US005781192A

United States Patent [19]

Kodimer

[11] Patent Number: 5,781,192

[45] Date of Patent: Jul. 14, 1998

[54] **DATA TRANSFER SYSTEM**

[75] Inventor: **Marianne L. Kodimer**, Anaheim, Calif.

[73] Assignee: **Canon Information Systems, Inc.**, Irvine, Calif.

[21] Appl. No.: **586,092**

[22] Filed: **Jan. 16, 1996**

[51] **Int. Cl.**[6] **G06F 3/00**

[52] **U.S. Cl.** **345/340**; 345/346

[58] **Field of Search** 395/340, 339, 395/343, 326; 345/340, 328, 329, 341, 342, 343, 344, 346, 347, 348, 352, 353, 359; 707/531

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,975,890	12/1990	Torres	345/340
5,070,478	12/1991	Abbott	707/531
5,430,836	7/1995	Wolf et al.	345/340
5,519,862	5/1996	Schaeffer et al.	395/701
5,548,740	8/1996	Kiyohara	711/100

OTHER PUBLICATIONS

IBM Technical Disclosure Bulletin, vol. 37, No. 3, Mar., 1994, pp. 325–326, "Cascade Clipboards: Accessing Multiple Clipboards".

IBM Technical Disclosure Bulletin, vol. 36, No. 6B, Jun., 1993, pp. 271–272, "Append Feature for Clipboards".

Research Disclosure, No. 3347, Mar., 1993, p. 200, "Enhancing the Single Level Clipboard to Provide Multiple Layers Graphically".

F. Davis, "The Windows 3.1 Bible", Peachpit Press, Inc., 1993, pp. 26–27, 68–70, 264–265, 301–304, 315–316, 497–500, 505, 508, 525–526, 745–748 and 752.

C. Petzold, "Programming Windows 3.1", Microsoft Press, 1992, Chapter 16, pp. 791–822.

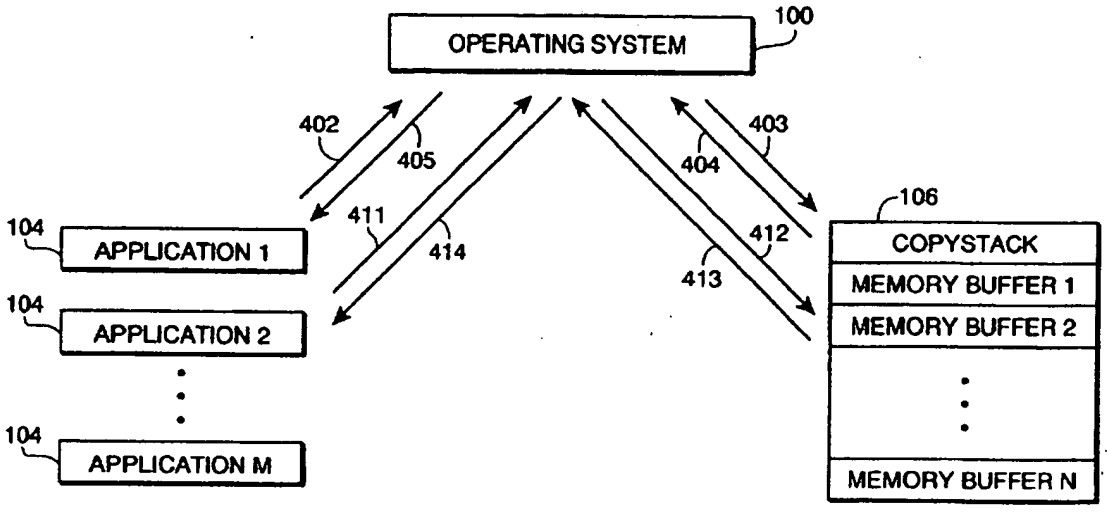
Simpson, *Mastering WordPerfect ® 5.1 & 5.2 for Windows*, Alameda, Ca., 1993.

Primary Examiner—Matthew M. Kim
Assistant Examiner—Steven P. Sax
Attorney, Agent, or Firm—Fitzpatrick, Cella, Harper & Scinto

[57] **ABSTRACT**

A computer implemented method for storing data from an instance of an application program into at least one of a plurality of buffers includes the steps of bringing an application program into focus, selecting data in the application program, selecting a store operation such as a "cut" or a "copy" operation while the application program is in focus, selecting one of the buffers while the application program is still in focus, and automatically storing the selected data into the selected one of the buffers. In a similar manner, data can be recalled from the buffers, such as in a "paste" operation, and any one of the buffers selected, without ever leaving focus of the application to which the data is recalled.

81 Claims, 26 Drawing Sheets



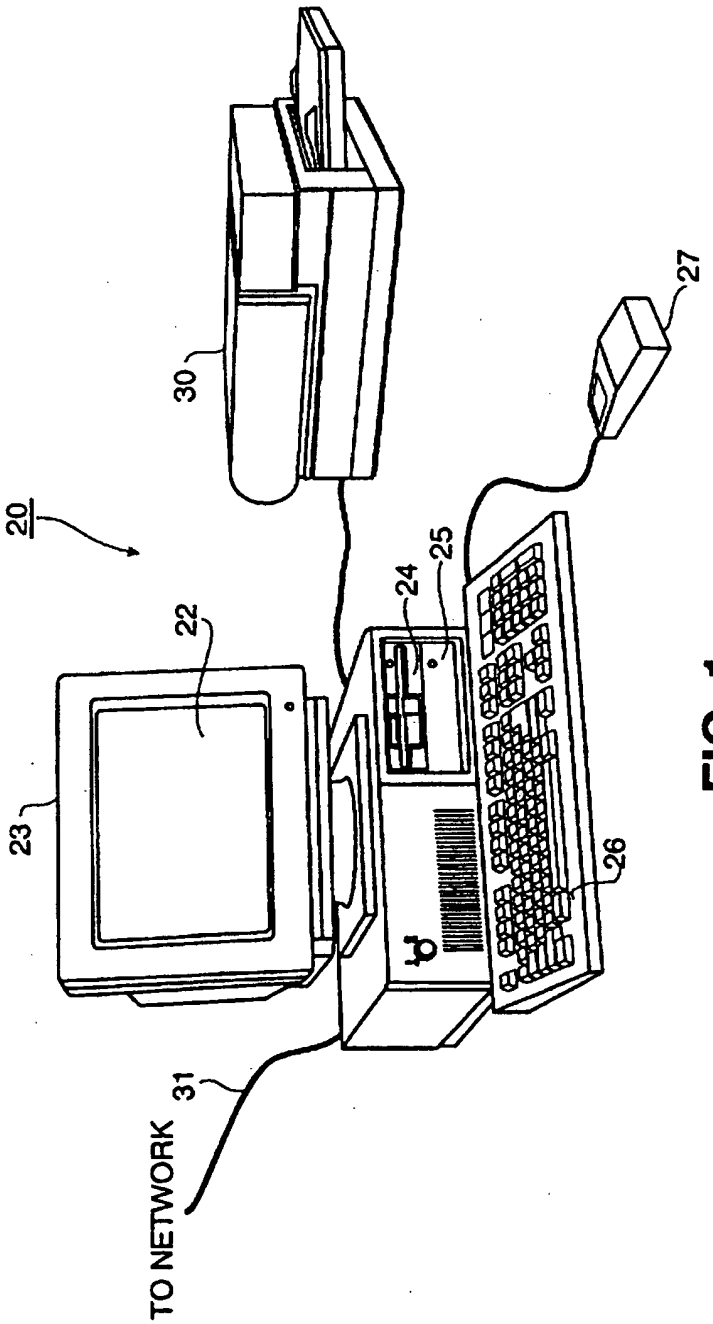


FIG. 1

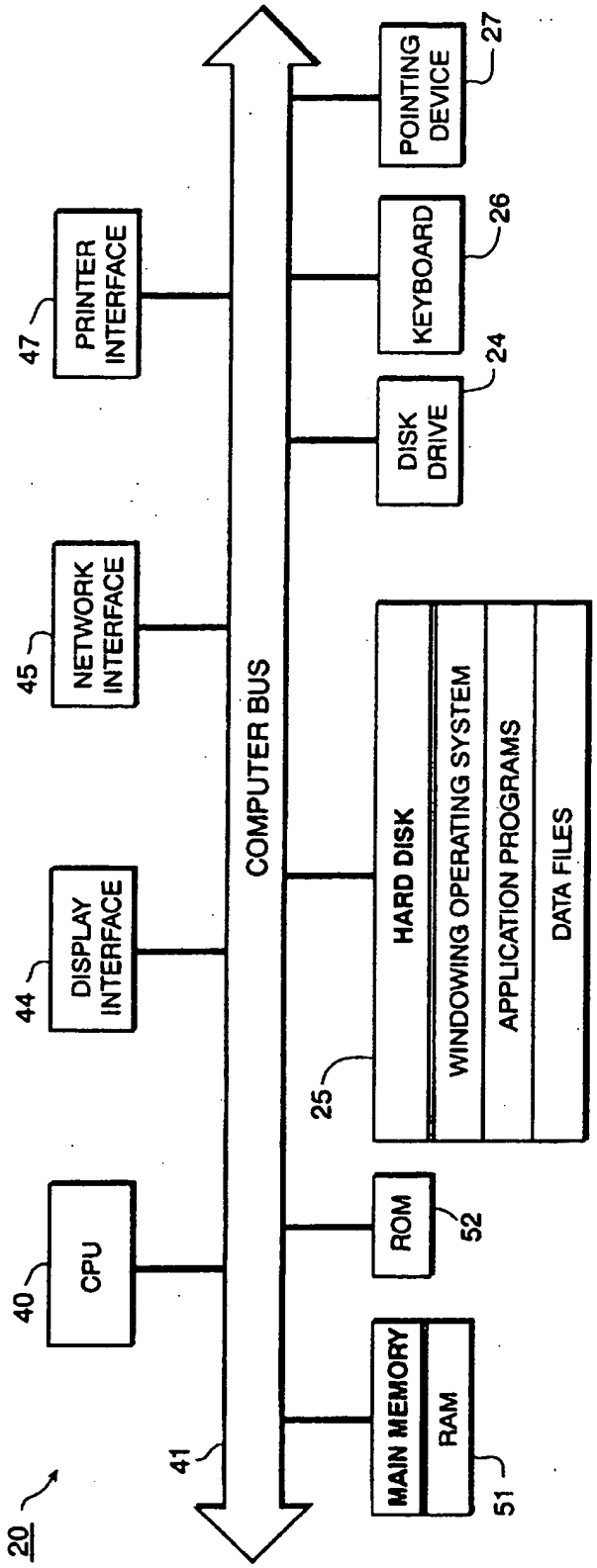

20
41
COMPUTER BUS
CPU
40
DISPLAY INTERFACE
44
NETWORK INTERFACE
45
PRINTER INTERFACE
47
MAIN MEMORY
RAM
51
ROM
52
HARD DISK
25
WINDOWING OPERATING SYSTEM
APPLICATION PROGRAMS
DATA FILES
DISK DRIVE
24
KEYBOARD
26
POINTING DEVICE
27

FIG. 2

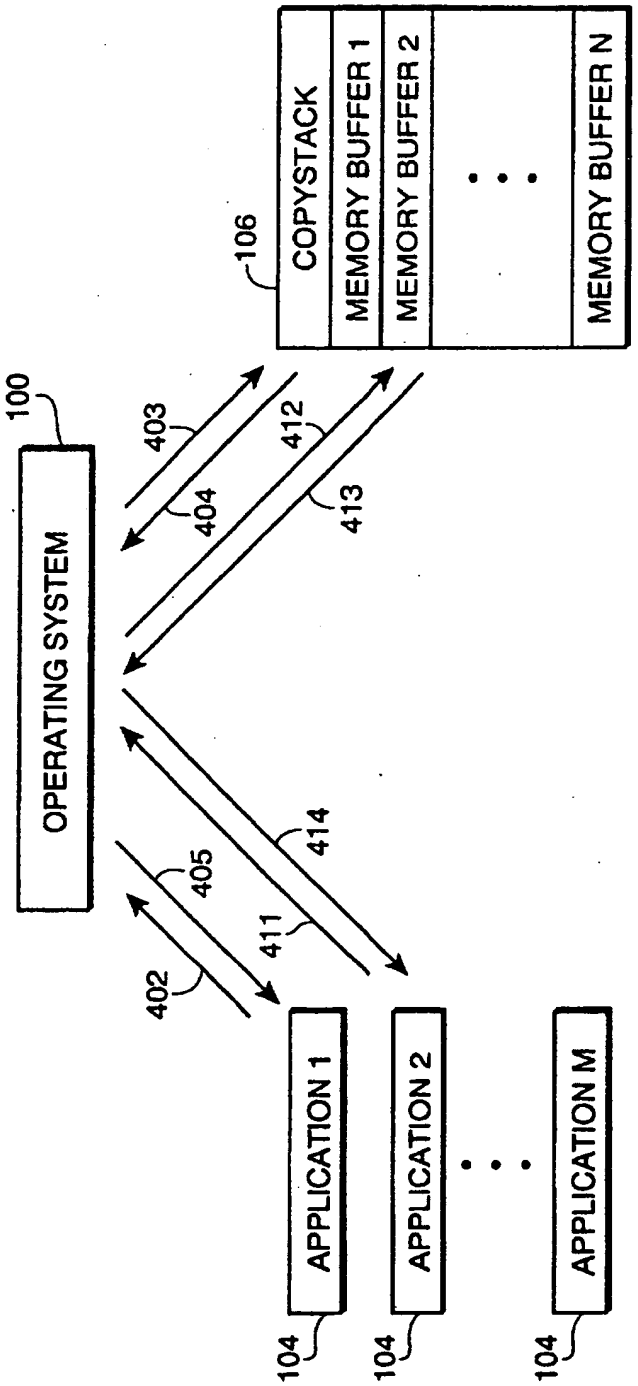

COPYSTACK
MEMORY BUFFER 1
MEMORY BUFFER 2
MEMORY BUFFER N
106
OPERATING SYSTEM
100
403
404
412
413
414
405
411
402
APPLICATION 1
APPLICATION 2
APPLICATION M
104
104
104

FIG. 3

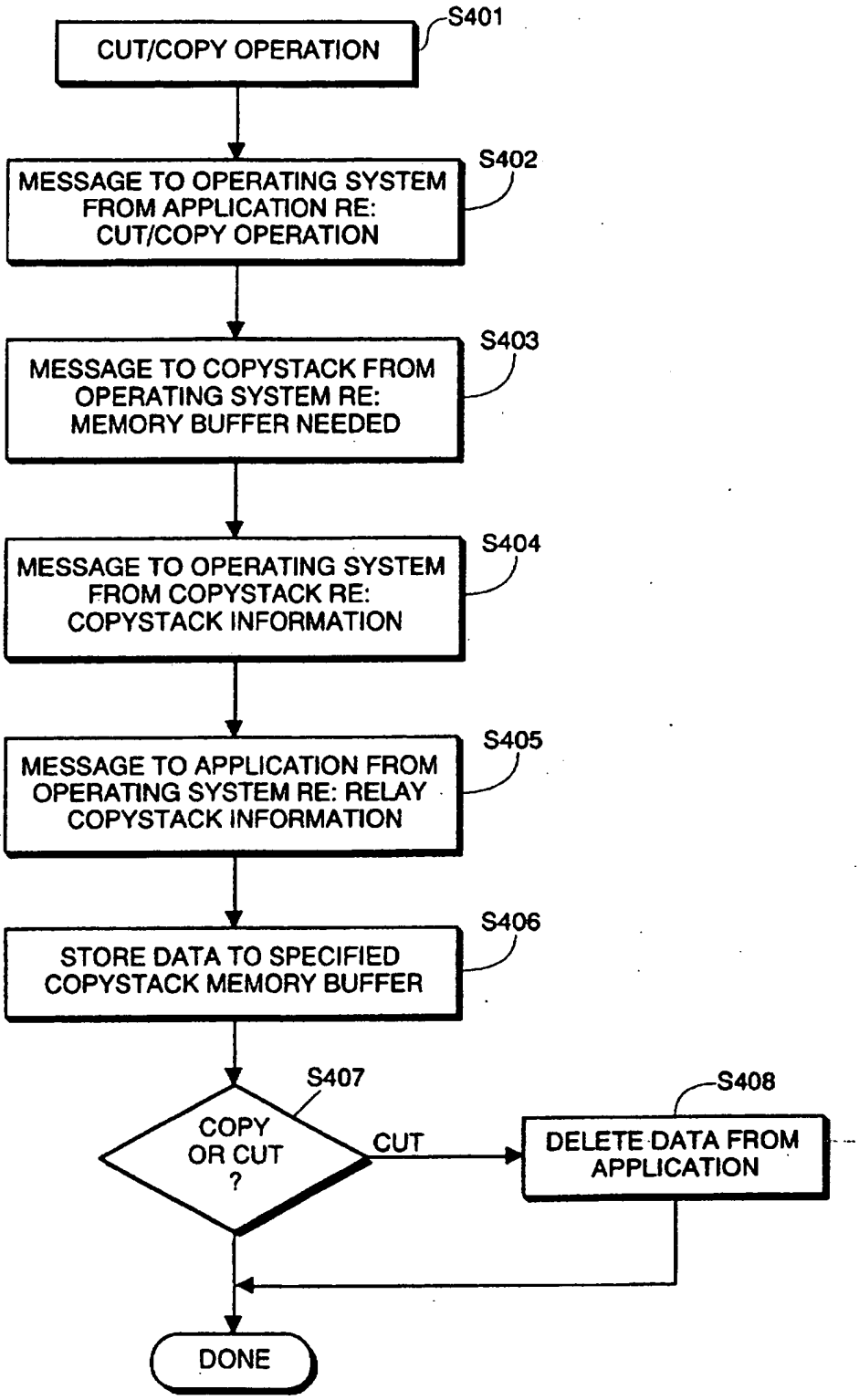

S401
CUT/COPY OPERATION
S402
MESSAGE TO OPERATING SYSTEM FROM APPLICATION RE: CUT/COPY OPERATION
S403
MESSAGE TO COPYSTACK FROM OPERATING SYSTEM RE: MEMORY BUFFER NEEDED
S404
MESSAGE TO OPERATING SYSTEM FROM COPYSTACK RE: COPYSTACK INFORMATION
S405
MESSAGE TO APPLICATION FROM OPERATING SYSTEM RE: RELAY COPYSTACK INFORMATION
S406
STORE DATA TO SPECIFIED COPYSTACK MEMORY BUFFER
S407
COPY OR CUT ?
CUT
S408
DELETE DATA FROM APPLICATION
DONE

FIG. 4(a)

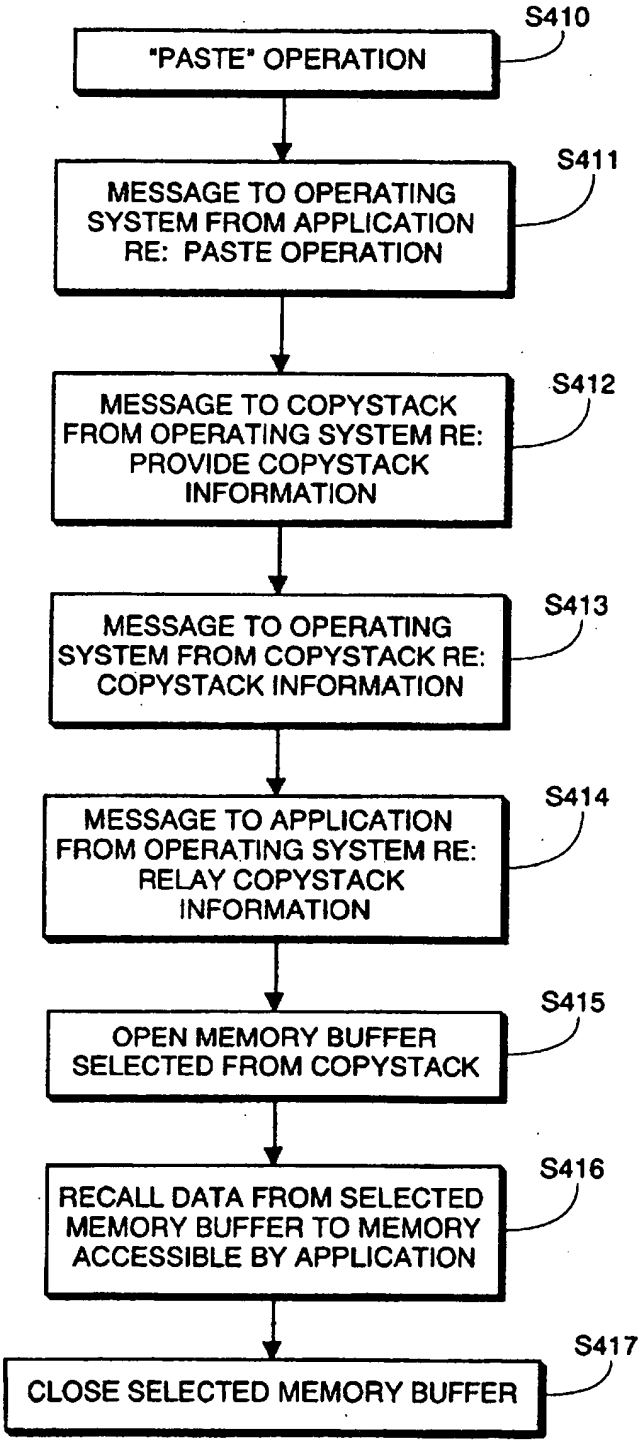

S410
"PASTE" OPERATION
S411
MESSAGE TO OPERATING SYSTEM FROM APPLICATION RE: PASTE OPERATION
S412
MESSAGE TO COPYSTACK FROM OPERATING SYSTEM RE: PROVIDE COPYSTACK INFORMATION
S413
MESSAGE TO OPERATING SYSTEM FROM COPYSTACK RE: COPYSTACK INFORMATION
S414
MESSAGE TO APPLICATION FROM OPERATING SYSTEM RE: RELAY COPYSTACK INFORMATION
S415
OPEN MEMORY BUFFER SELECTED FROM COPYSTACK
S416
RECALL DATA FROM SELECTED MEMORY BUFFER TO MEMORY ACCESSIBLE BY APPLICATION
S417
CLOSE SELECTED MEMORY BUFFER

FIG. 4(b)

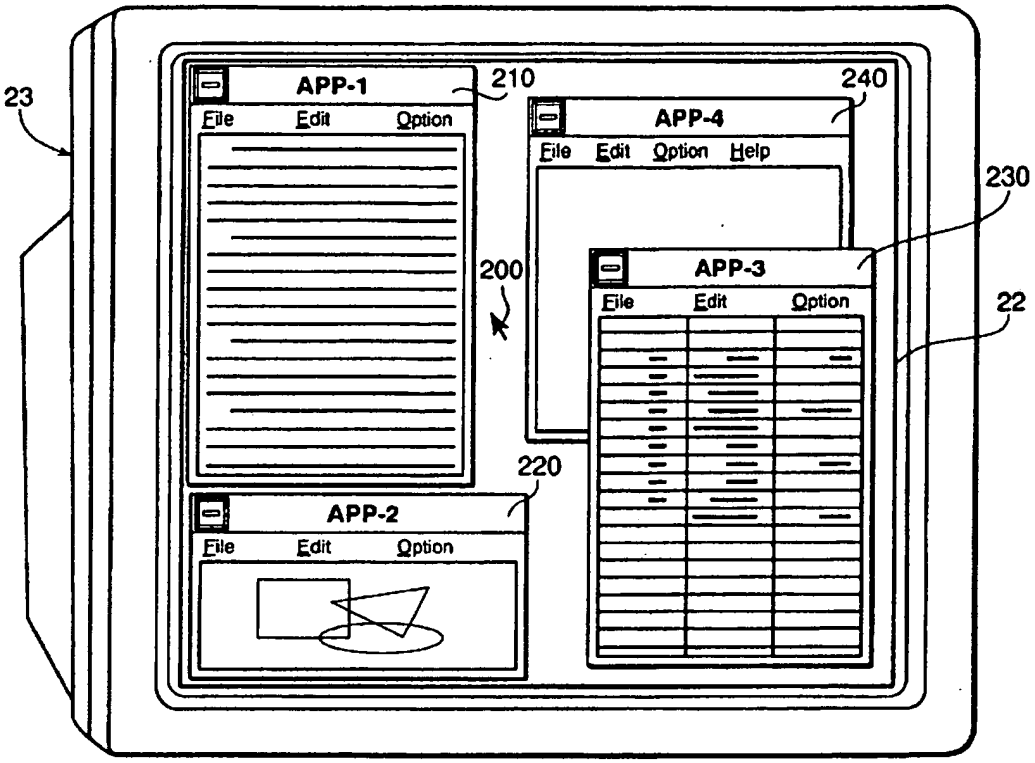

23
APP-1
File Edit Option
210
APP-4
File Edit Option Help
240
230
APP-3
File Edit Option
200
22
220
APP-2
File Edit Option

FIG. 5

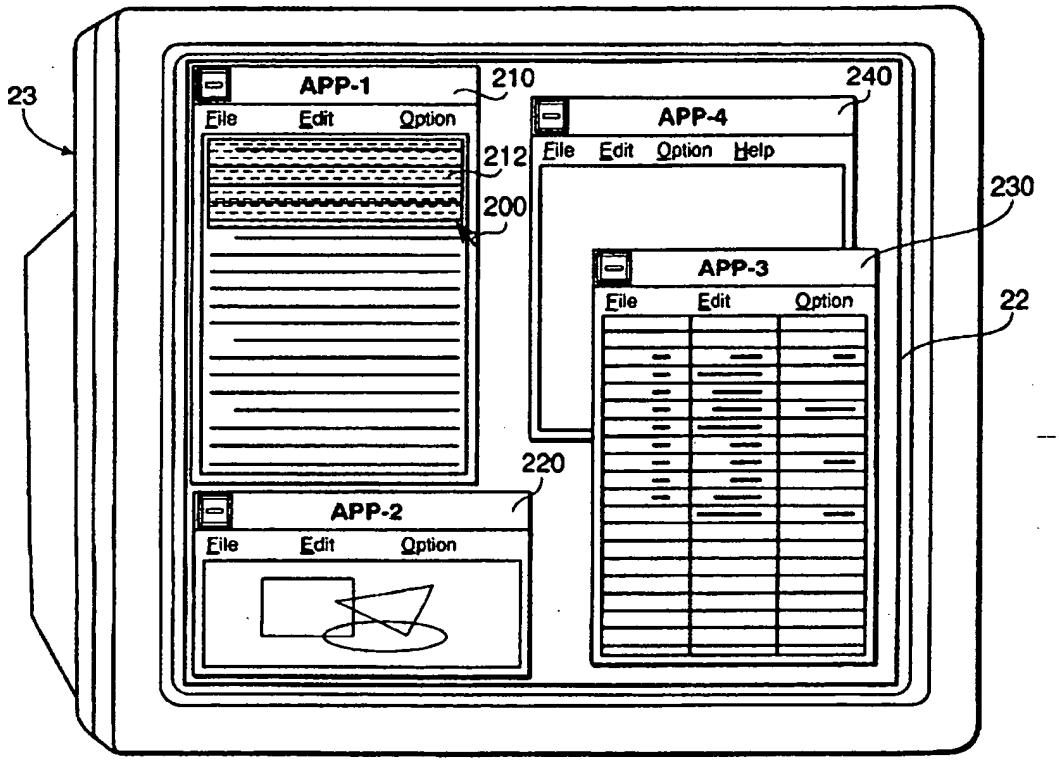

23
APP-1
File Edit Option
210
212
200
APP-4
File Edit Option Help
240
230
APP-3
File Edit Option
22
220
APP-2
File Edit Option

FIG. 6(a)

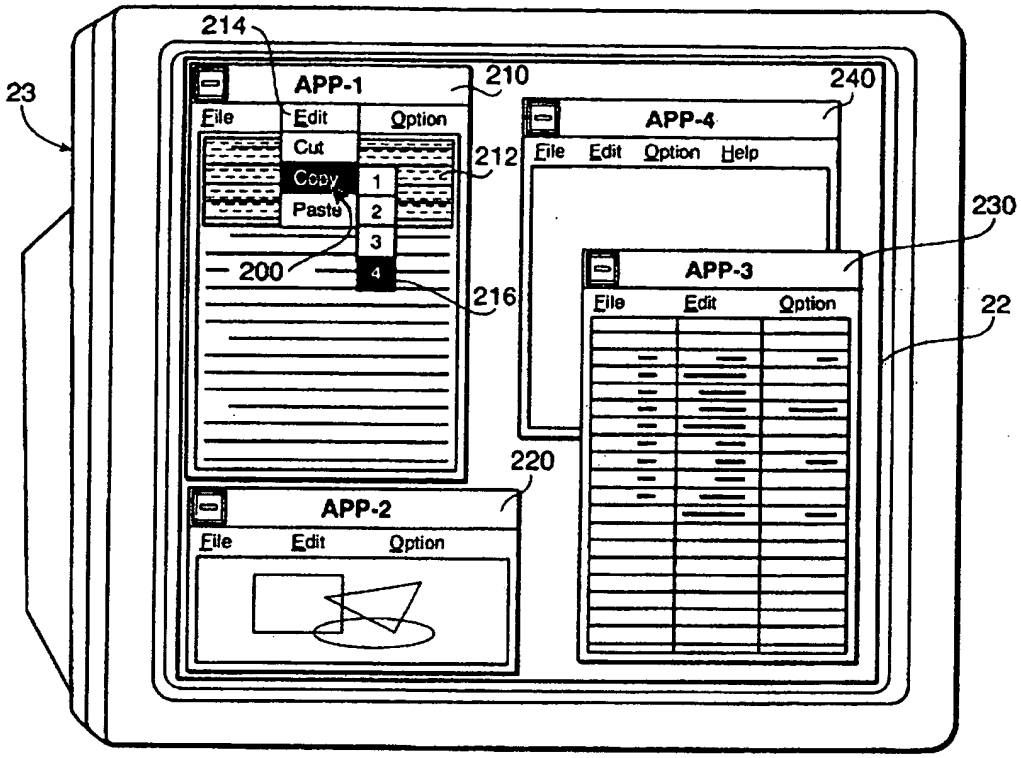

214
23
APP-1
210
240
File Edit Option
Cut
Copy
Paste
1
2
3
4
212
APP-4
File Edit Option Help
230
200
216
APP-3
File Edit Option
22
220
APP-2
File Edit Option

FIG. 6(b)

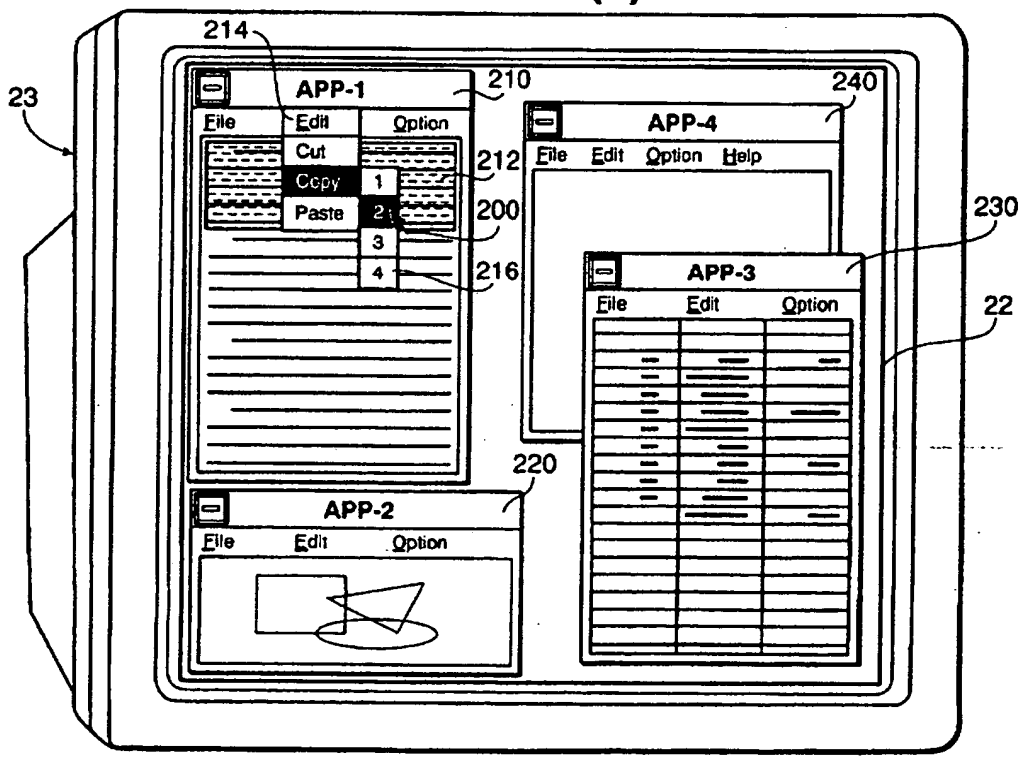

214
23
APP-1
210
240
File Edit Option
Cut
Copy
Paste
1
2
3
4
212
200
216
APP-4
File Edit Option Help
230
APP-3
File Edit Option
22
220
APP-2
File Edit Option

FIG. 6(c)

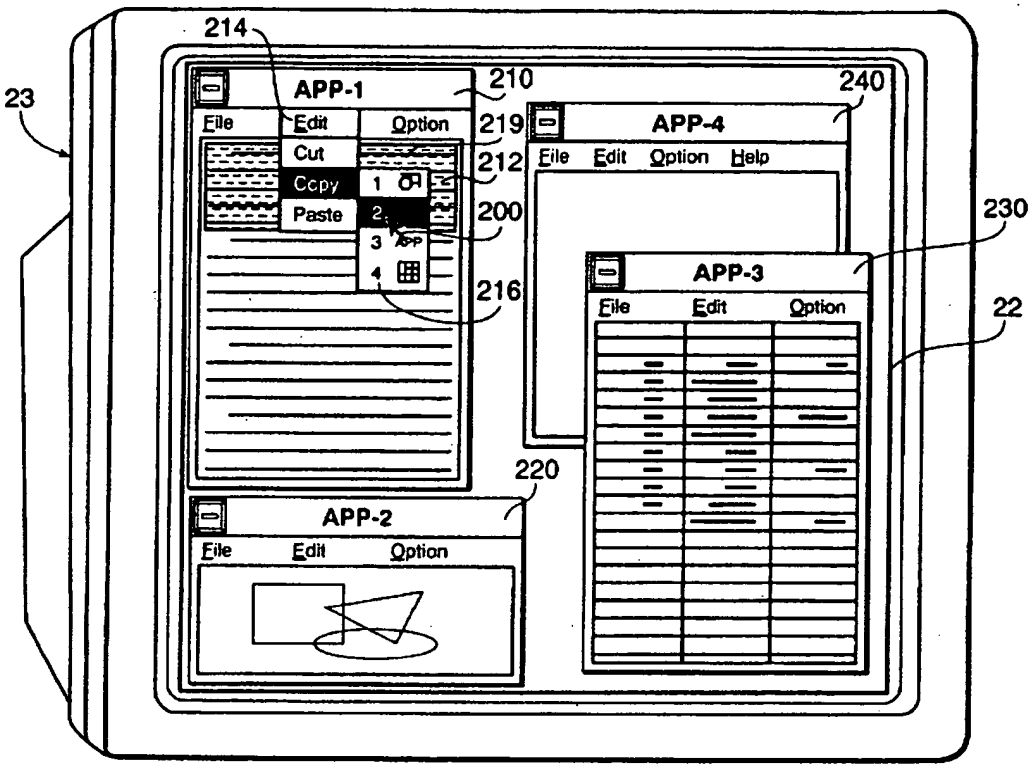

214
23
APP-1
210
240
File Edit Option
219
APP-4
Cut
212
File Edit Option Help
Copy
1
Paste
2
200
230
3
4
APP-3
216
File Edit Option
22
220
APP-2
File Edit Option

FIG. 6(d)

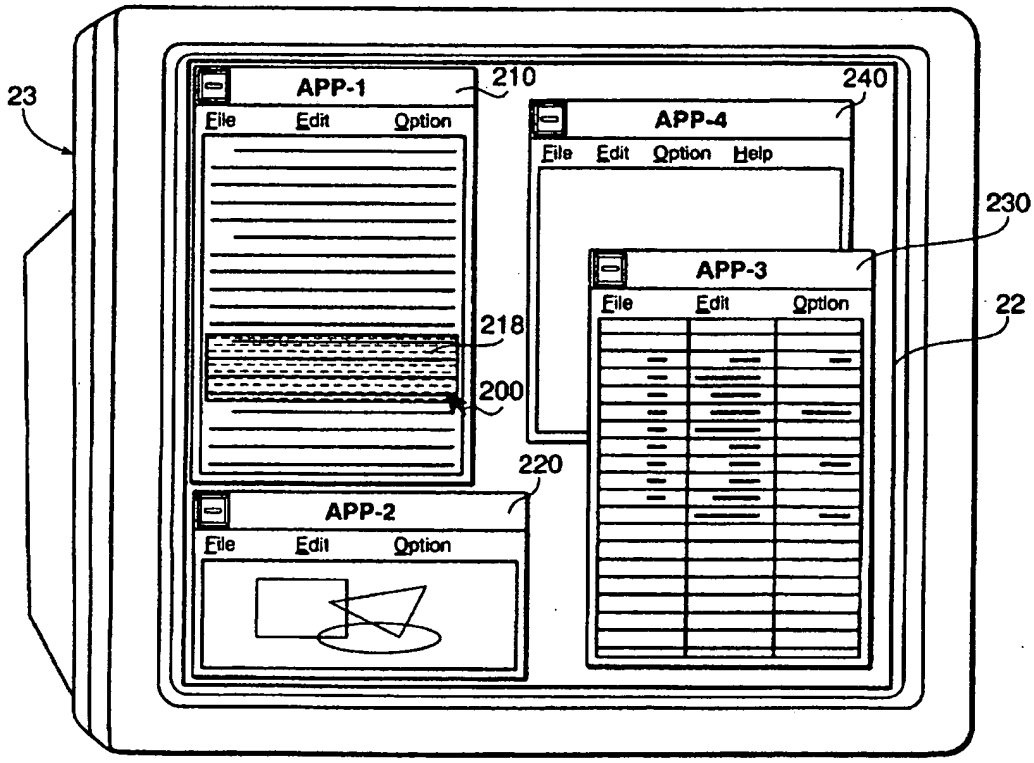

23
APP-1
210
240
File Edit Option
APP-4
File Edit Option Help
230
APP-3
218
File Edit Option
22
200
220
APP-2
File Edit Option

FIG. 7(a)

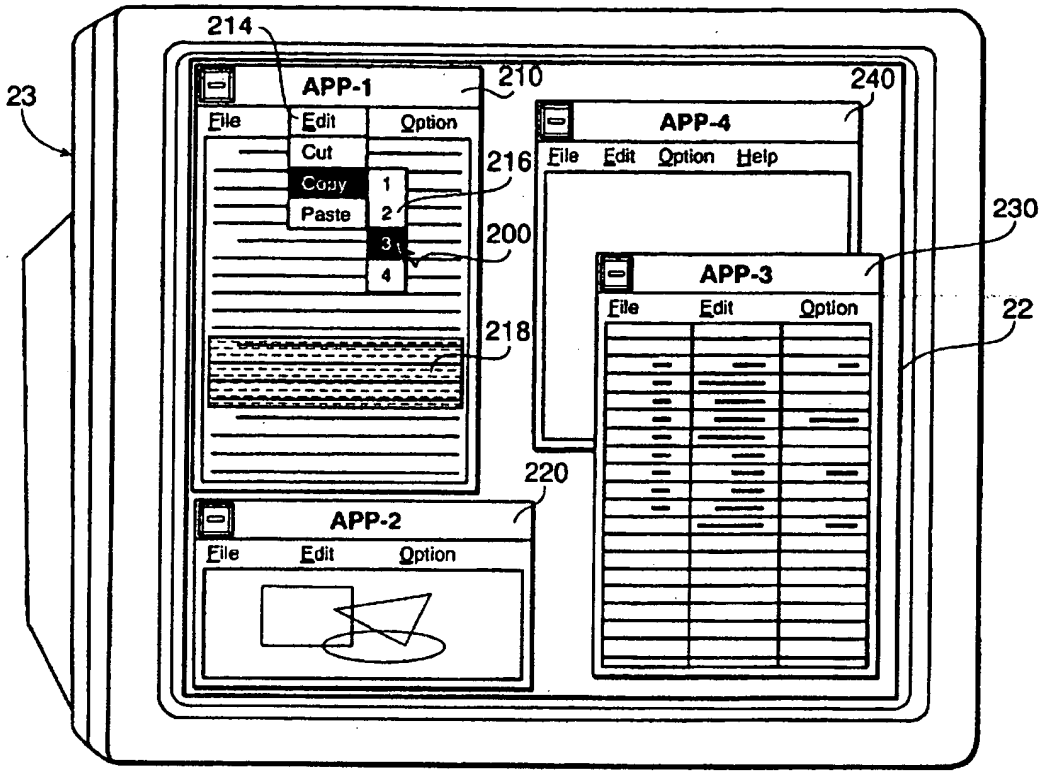

214
23
APP-1
210
240
File
Edit
Option
APP-4
Cut
Copy
Paste
1
2
3
4
216
200
File
Edit
Option
Help
230
APP-3
218
22
220
APP-2
File
Edit
Option

FIG. 7(b)

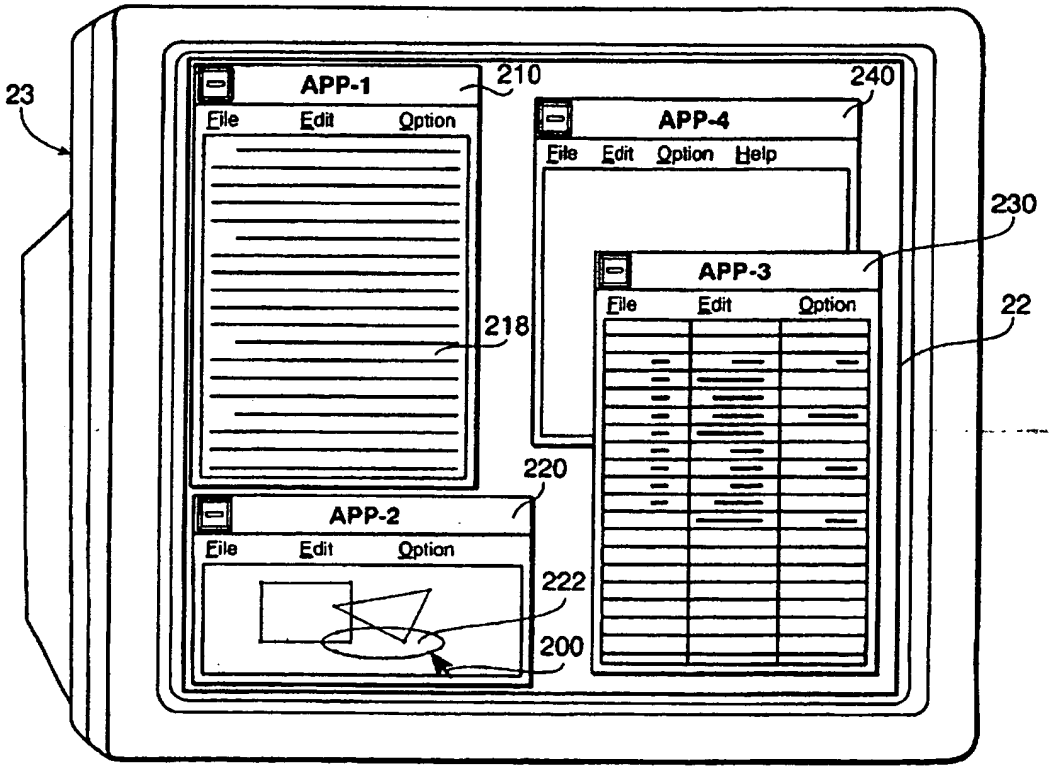

23
APP-1
210
240
File
Edit
Option
APP-4
File
Edit
Option
Help
230
APP-3
218
22
220
APP-2
File
Edit
Option
222
200

FIG. 8(a)

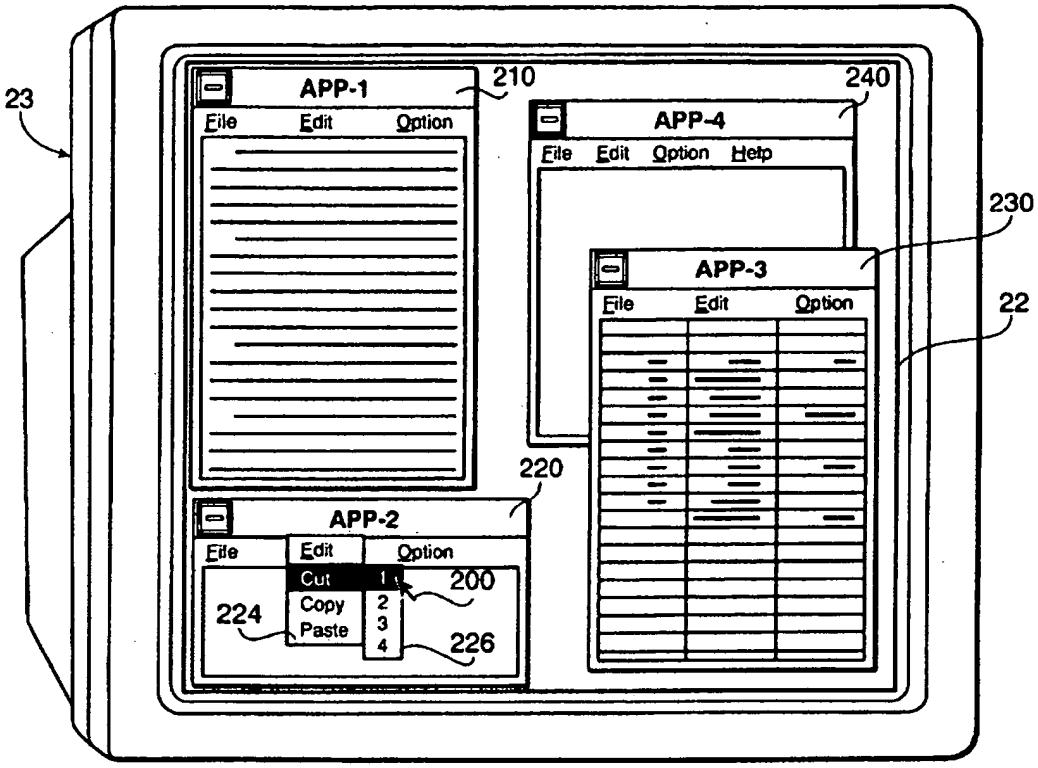

23
APP-1
File Edit Option
210
APP-4
File Edit Option Help
240
230
APP-3
File Edit Option
22
220
APP-2
File Edit Option
Cut
Copy
Paste
1
2
3
4
200
224
226

FIG. 8(b)

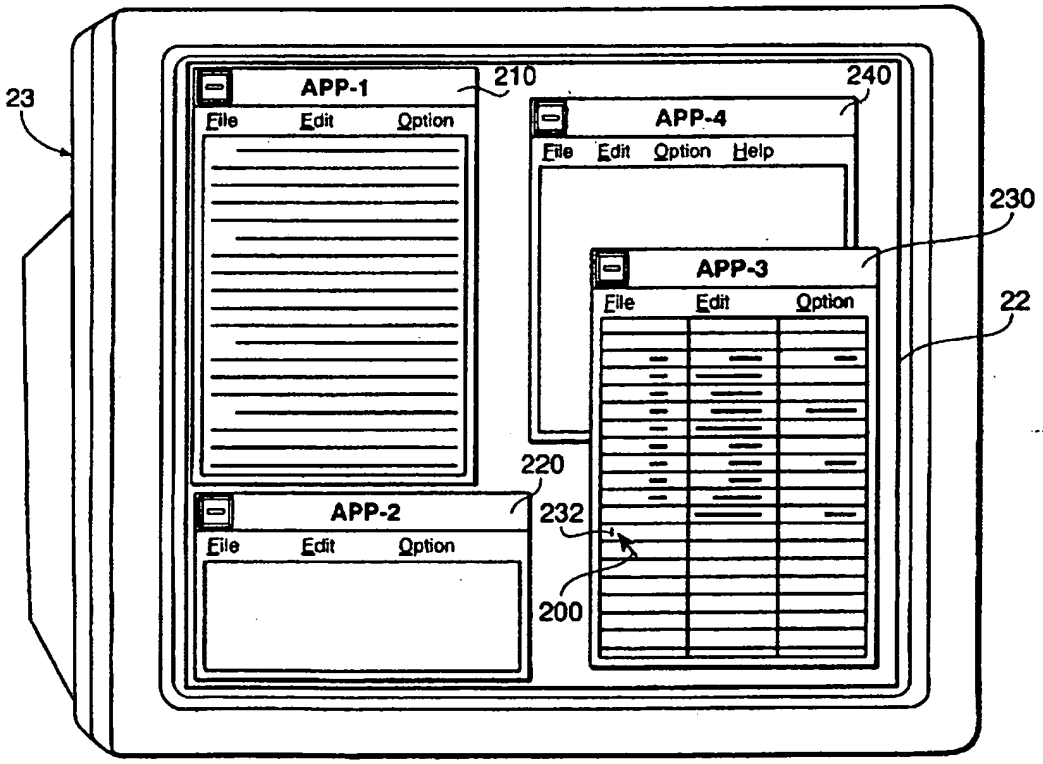

23
APP-1
File Edit Option
210
APP-4
File Edit Option Help
240
230
APP-3
File Edit Option
22
220
APP-2
File Edit Option
232
200

FIG. 9(a)

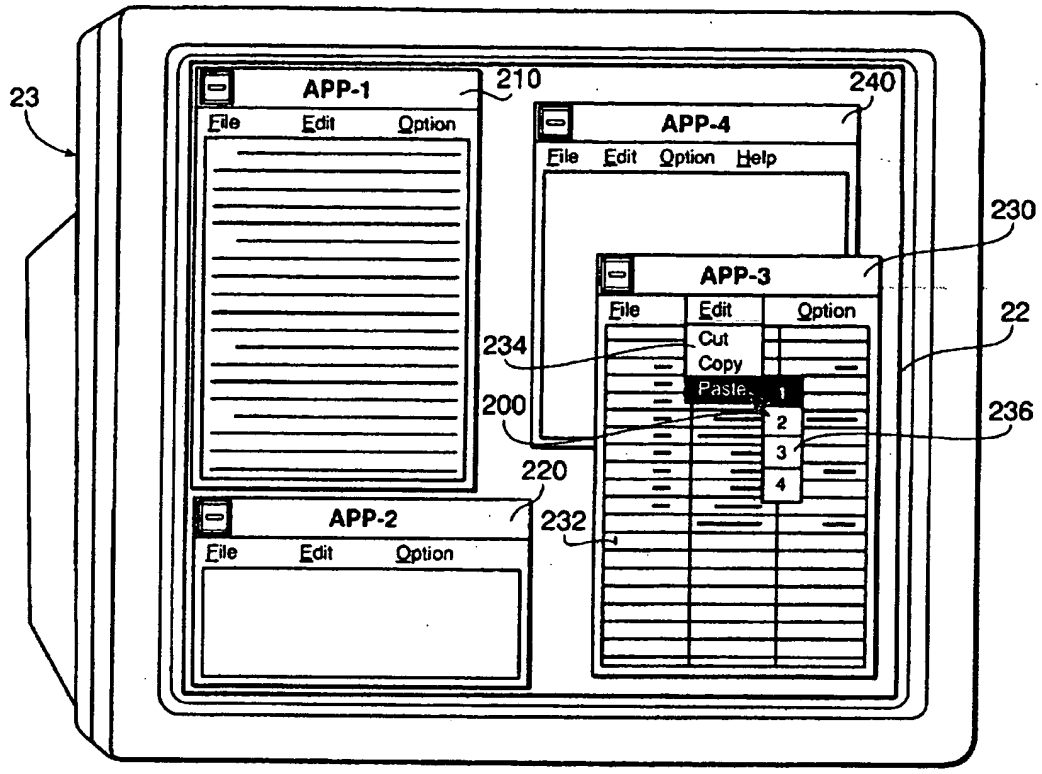

23
APP-1
File Edit Option
210
240
APP-4
File Edit Option Help
230
APP-3
File Edit Option
22
234
Cut
Copy
Paste
200
1
2
3
4
236
220
APP-2
File Edit Option
232

FIG. 9(b)

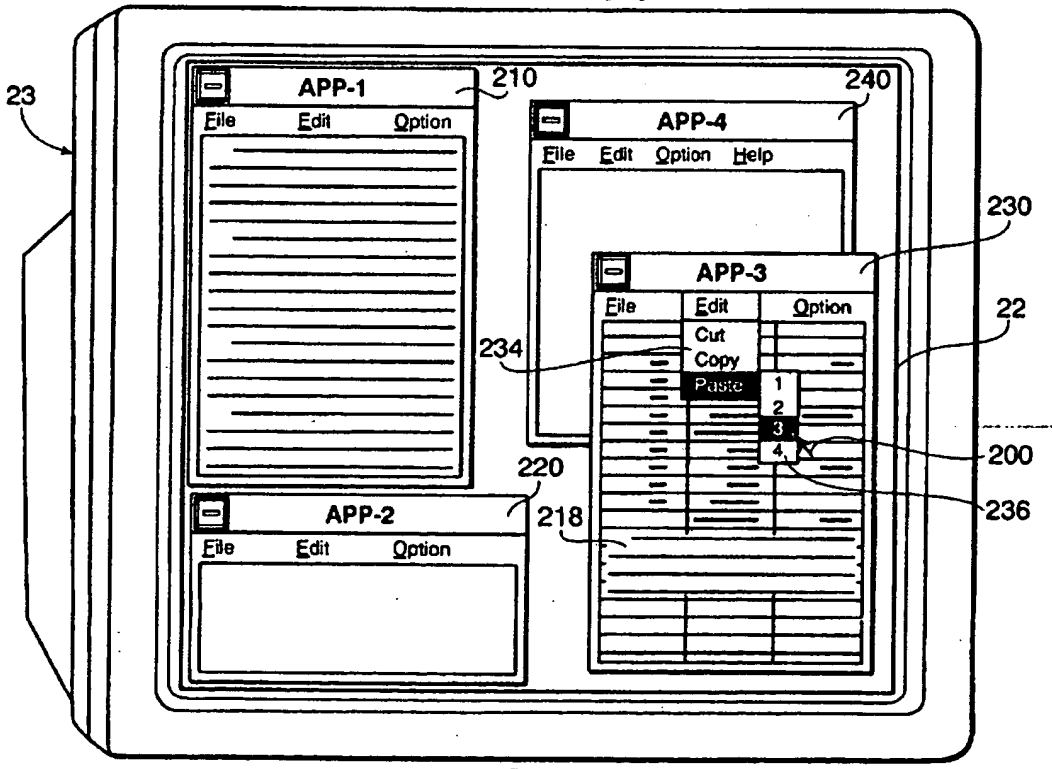

23
APP-1
File Edit Option
210
240
APP-4
File Edit Option Help
230
APP-3
File Edit Option
22
234
Cut
Copy
Paste
1
2
3
4
200
236
220
APP-2
File Edit Option
218

FIG. 9(c)

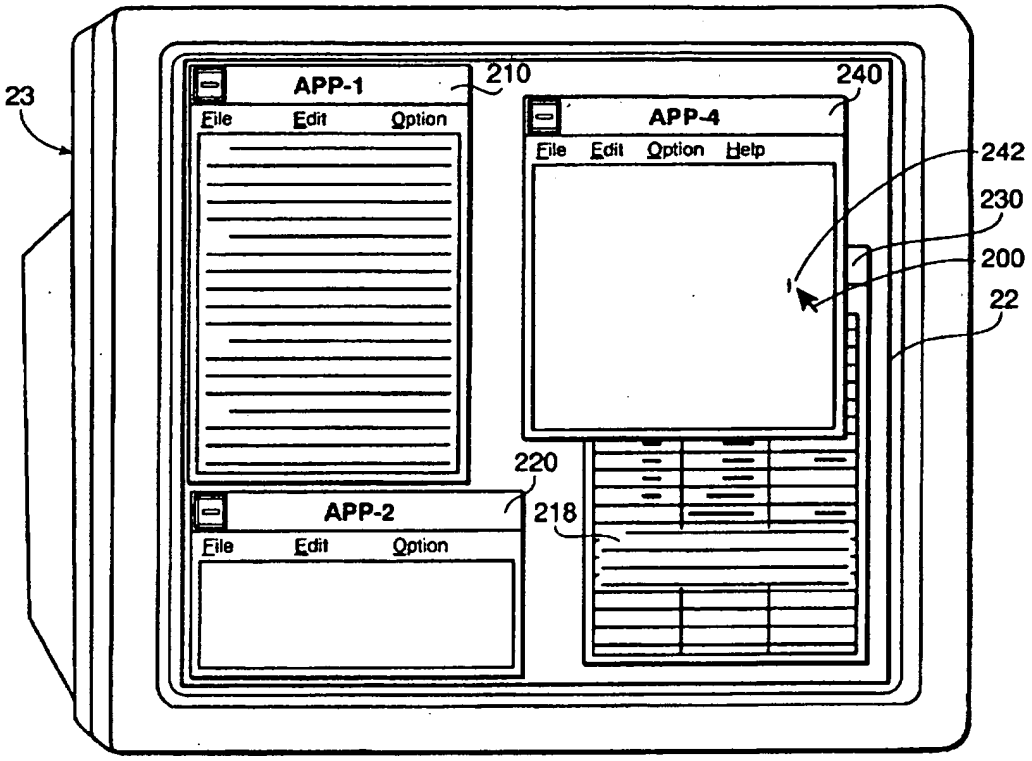


FIG. 10(a)

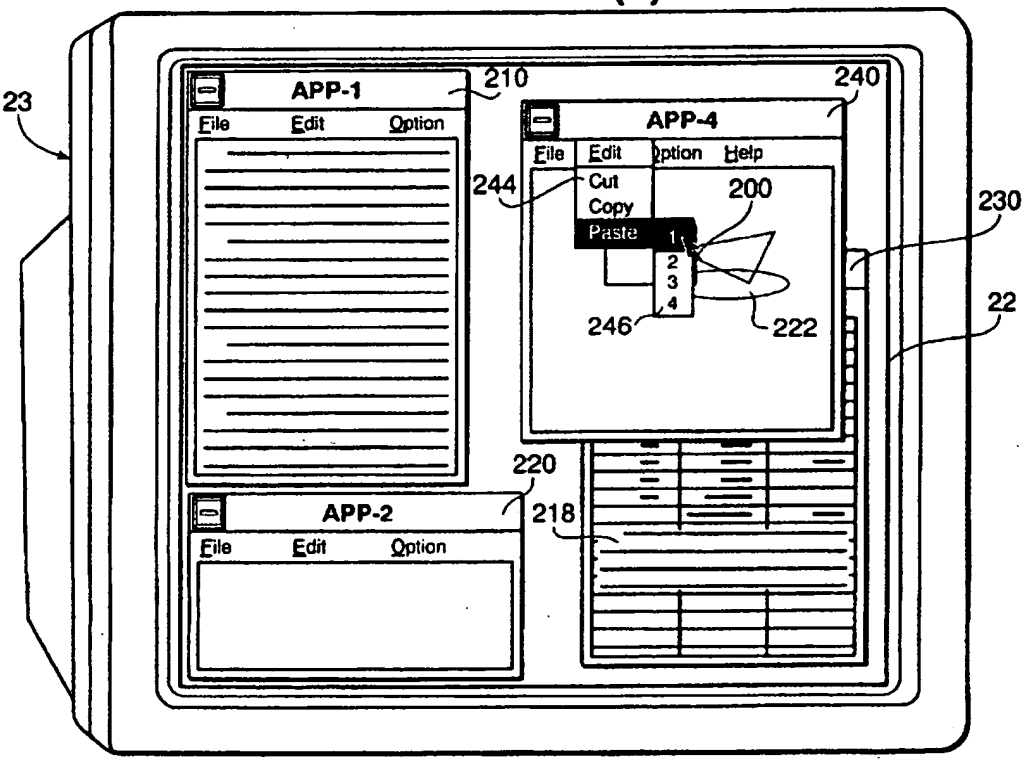


FIG. 10(b)

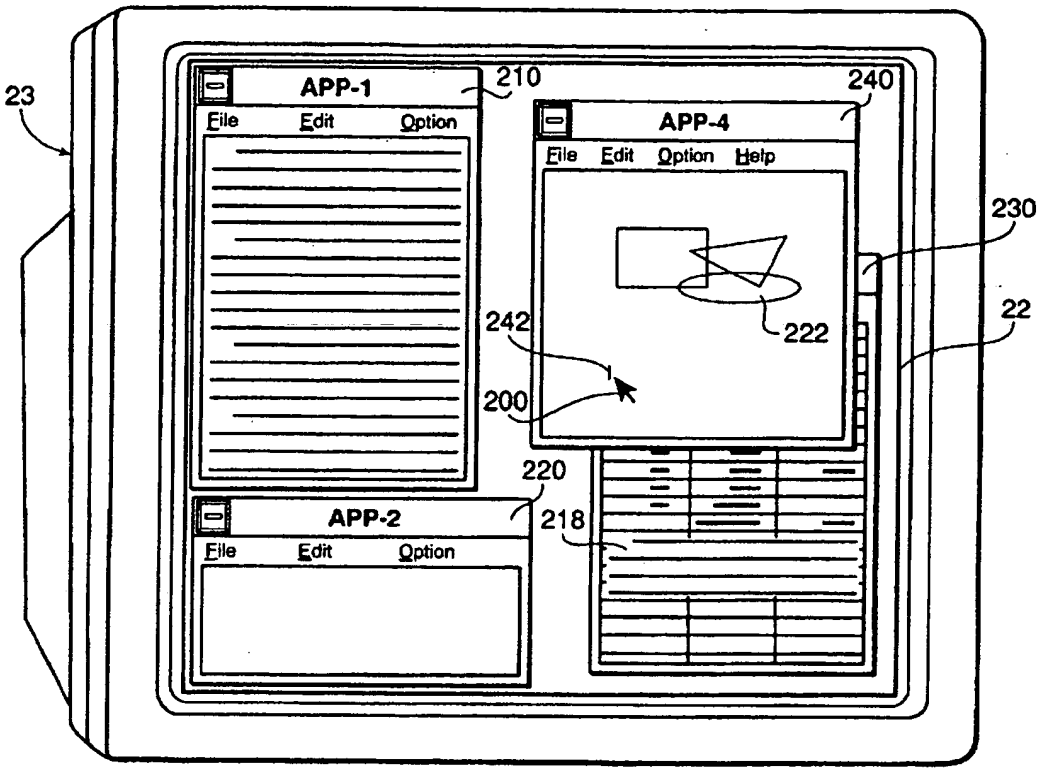


FIG. 11(a)

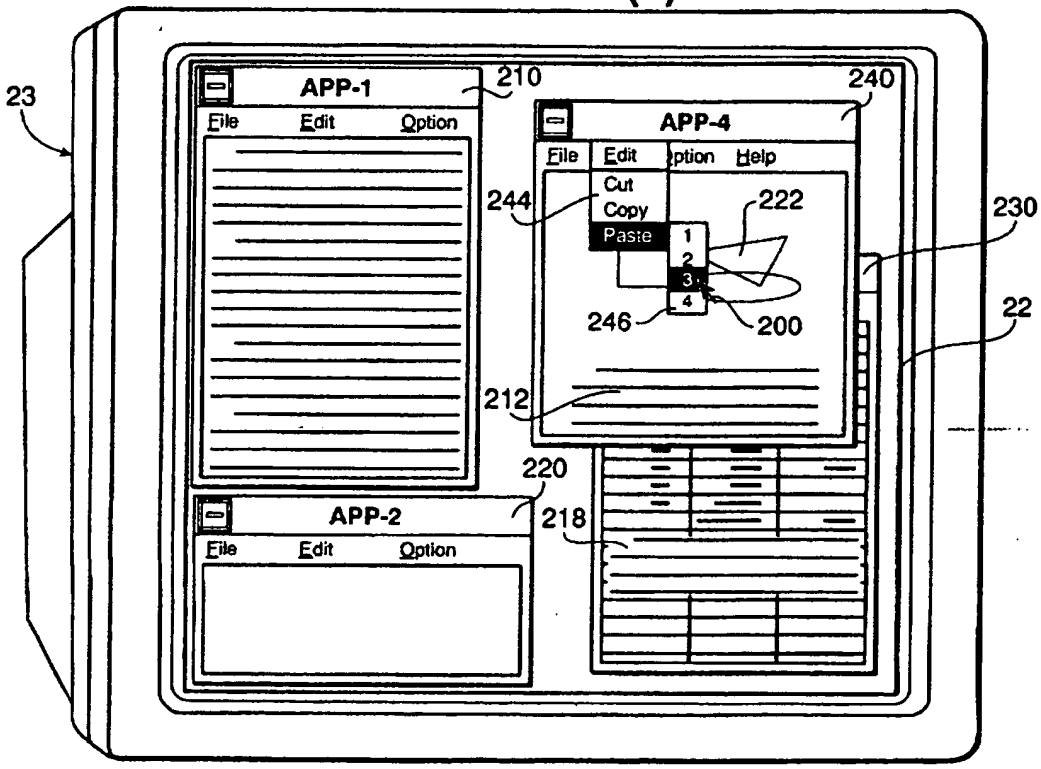


FIG. 11(b)

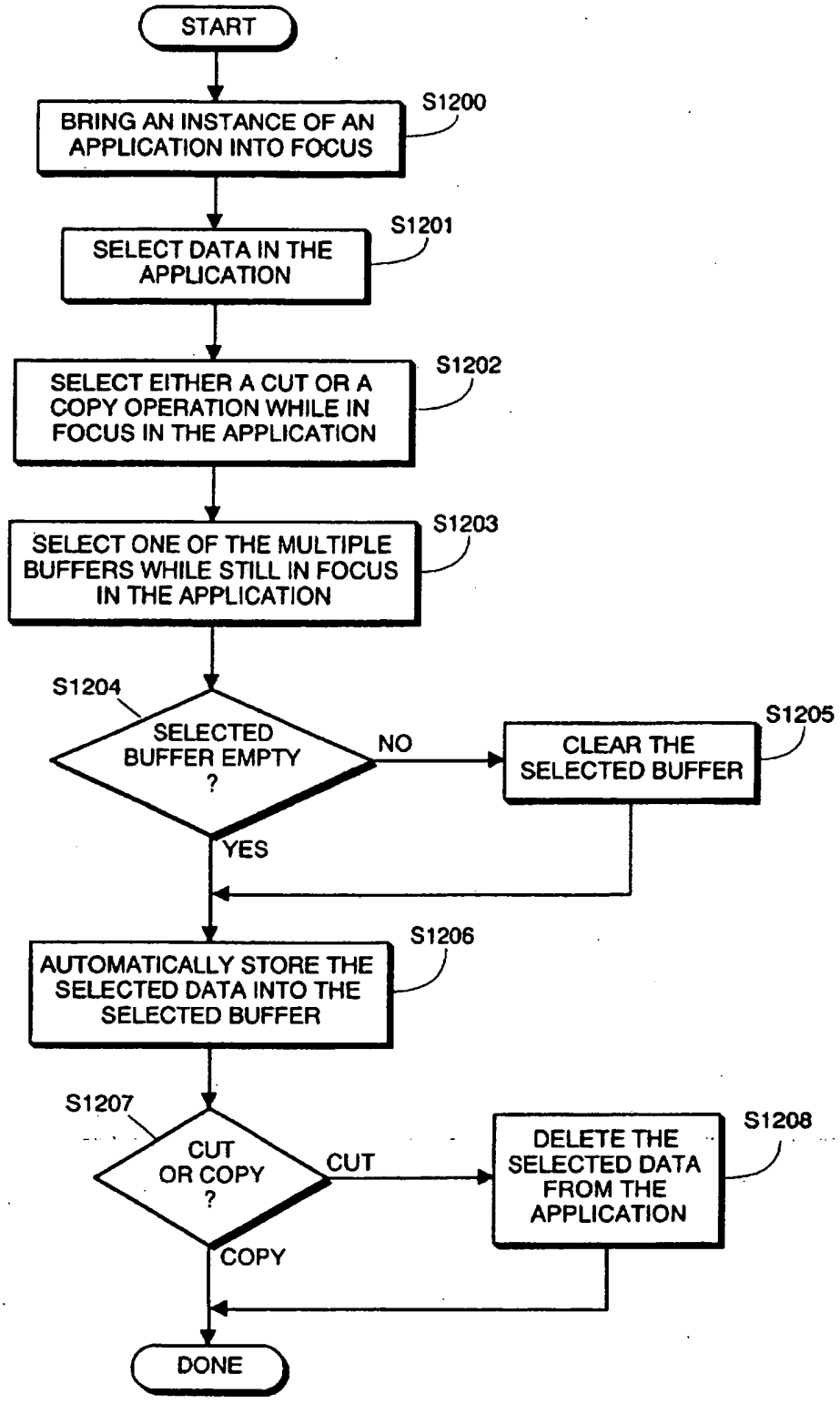

START
BRING AN INSTANCE OF AN APPLICATION INTO FOCUS
S1200
SELECT DATA IN THE APPLICATION
S1201
SELECT EITHER A CUT OR A COPY OPERATION WHILE IN FOCUS IN THE APPLICATION
S1202
SELECT ONE OF THE MULTIPLE BUFFERS WHILE STILL IN FOCUS IN THE APPLICATION
S1203
S1204
SELECTED BUFFER EMPTY ?
NO
CLEAR THE SELECTED BUFFER
S1205
YES
AUTOMATICALLY STORE THE SELECTED DATA INTO THE SELECTED BUFFER
S1206
S1207
CUT OR COPY ?
CUT
DELETE THE SELECTED DATA FROM THE APPLICATION
S1208
COPY
DONE

FIG. 12(a)

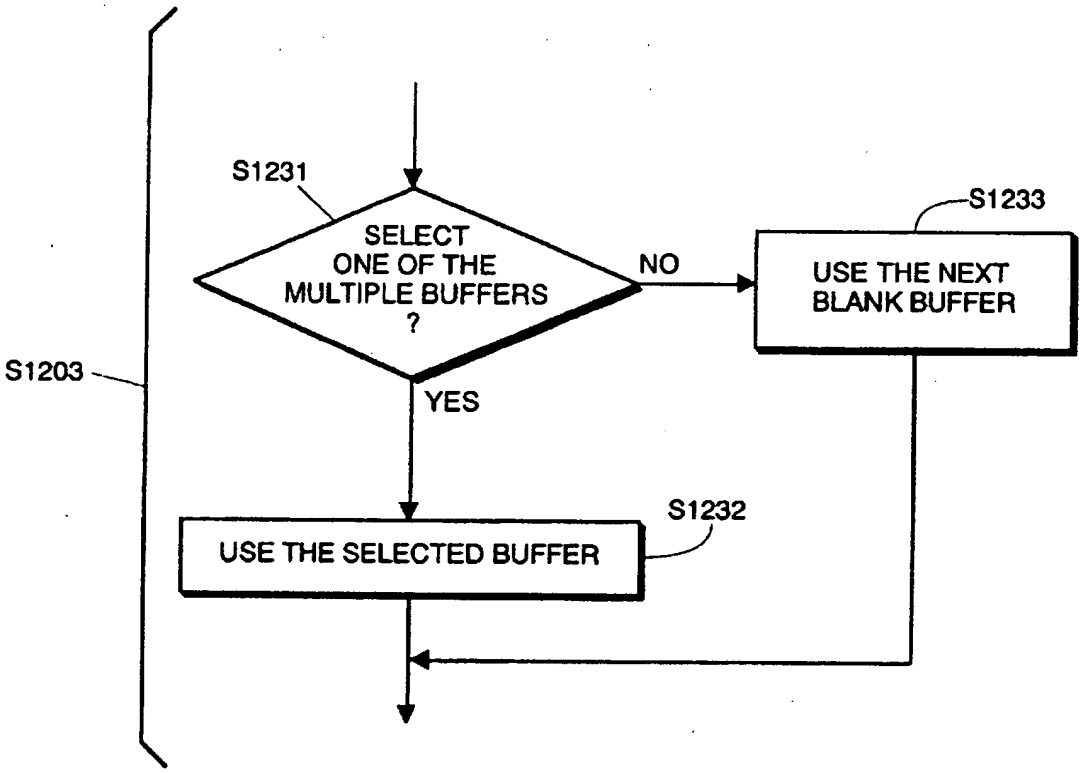

S1231
SELECT ONE OF THE MULTIPLE BUFFERS ?
NO
S1233
USE THE NEXT BLANK BUFFER
S1203
YES
S1232
USE THE SELECTED BUFFER

FIG. 12(b)

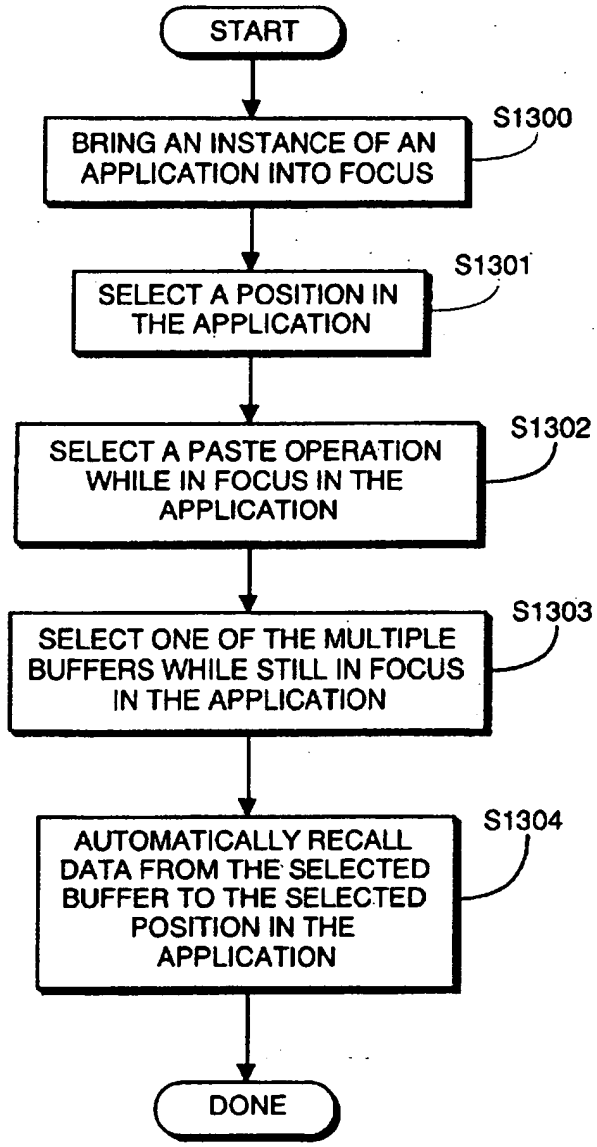

START
BRING AN INSTANCE OF AN APPLICATION INTO FOCUS
S1300
SELECT A POSITION IN THE APPLICATION
S1301
SELECT A PASTE OPERATION WHILE IN FOCUS IN THE APPLICATION
S1302
SELECT ONE OF THE MULTIPLE BUFFERS WHILE STILL IN FOCUS IN THE APPLICATION
S1303
AUTOMATICALLY RECALL DATA FROM THE SELECTED BUFFER TO THE SELECTED POSITION IN THE APPLICATION
S1304
DONE

FIG. 13(a)

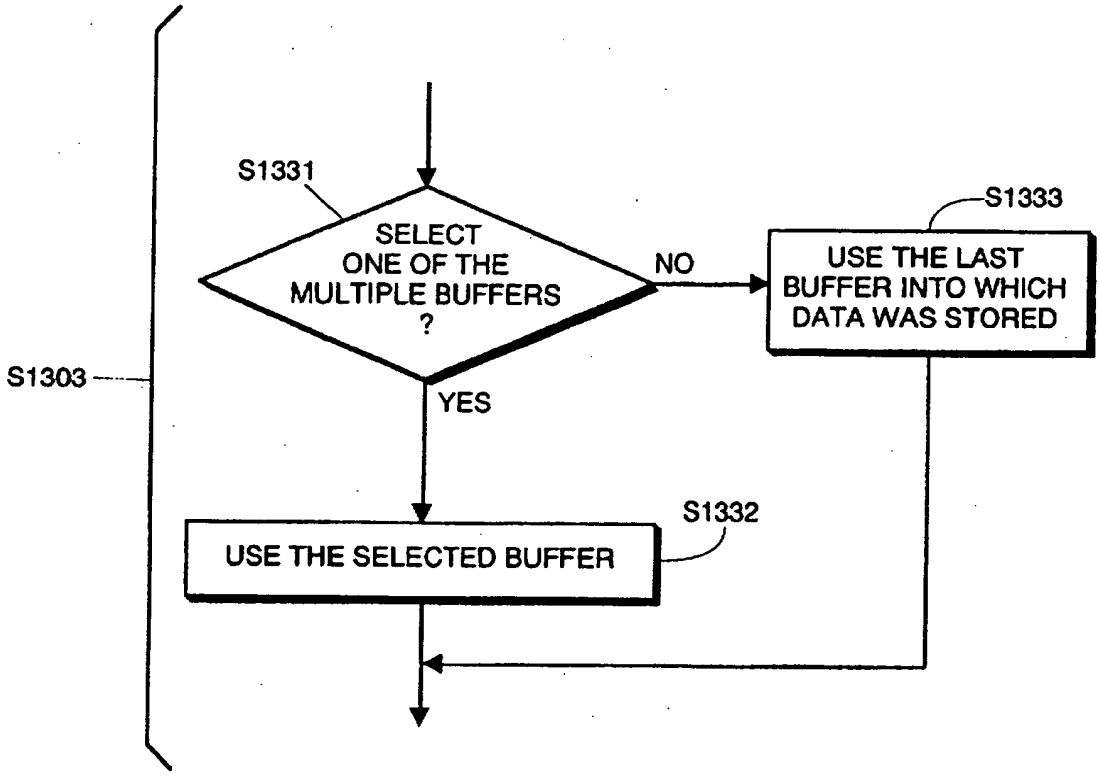

S1303
S1331
SELECT ONE OF THE MULTIPLE BUFFERS ?
NO
YES
S1333
USE THE LAST BUFFER INTO WHICH DATA WAS STORED
S1332
USE THE SELECTED BUFFER

FIG. 13(b)

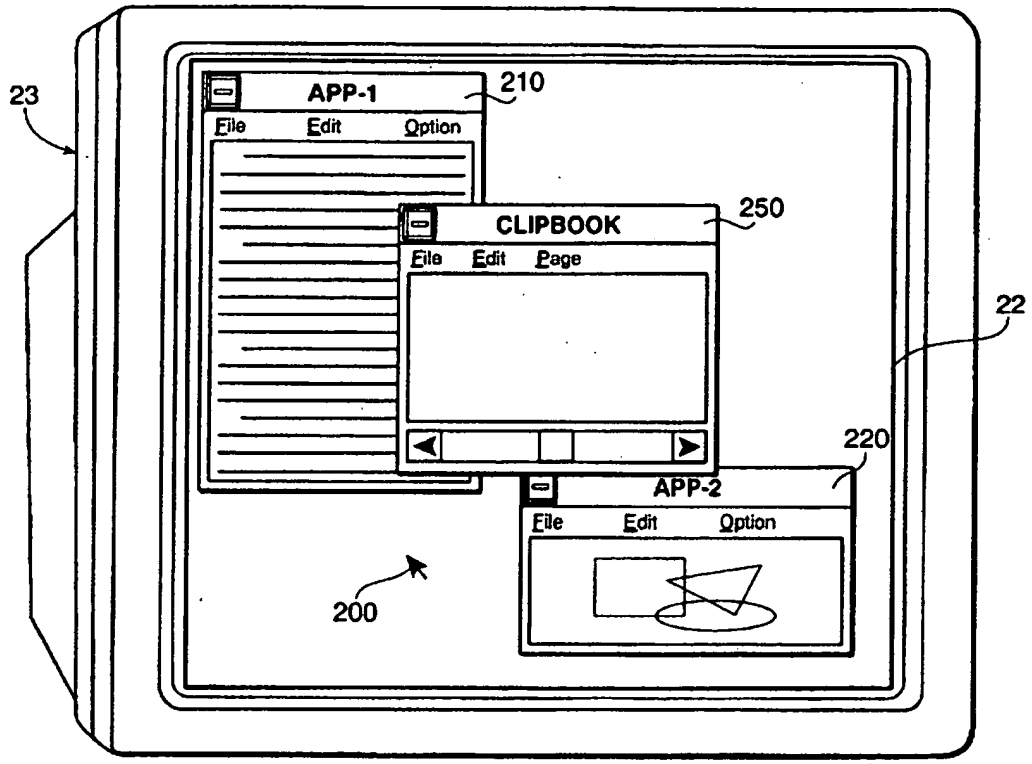


FIG. 14(a) PRIOR ART

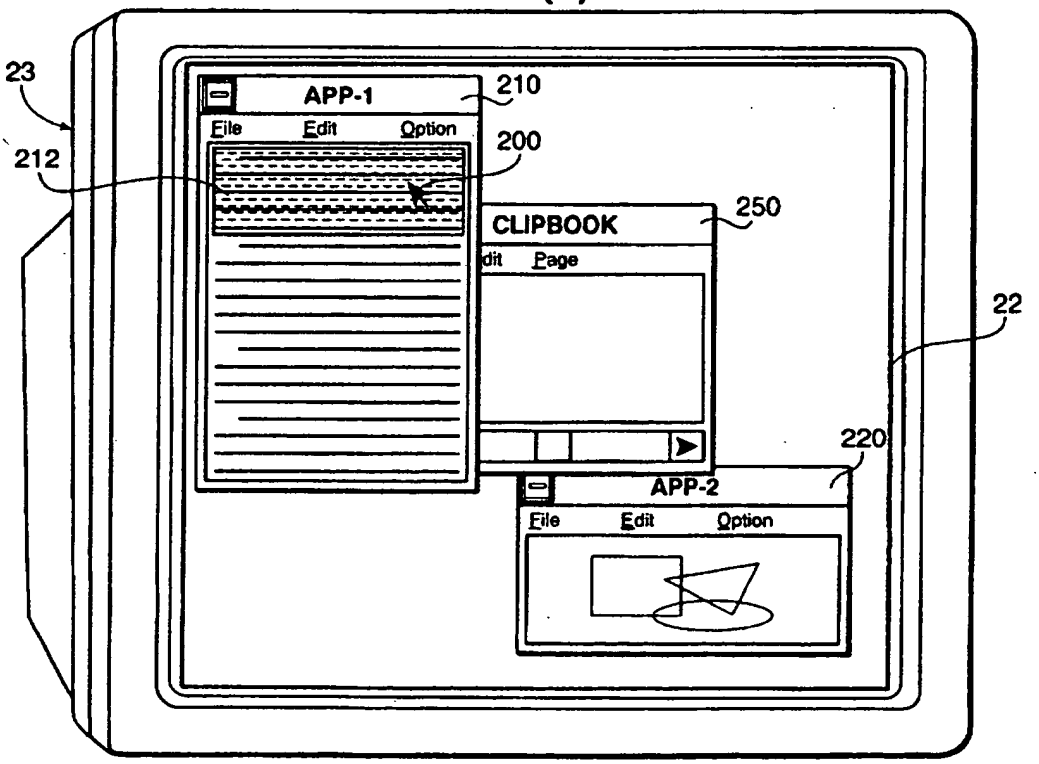


FIG. 14(b) PRIOR ART

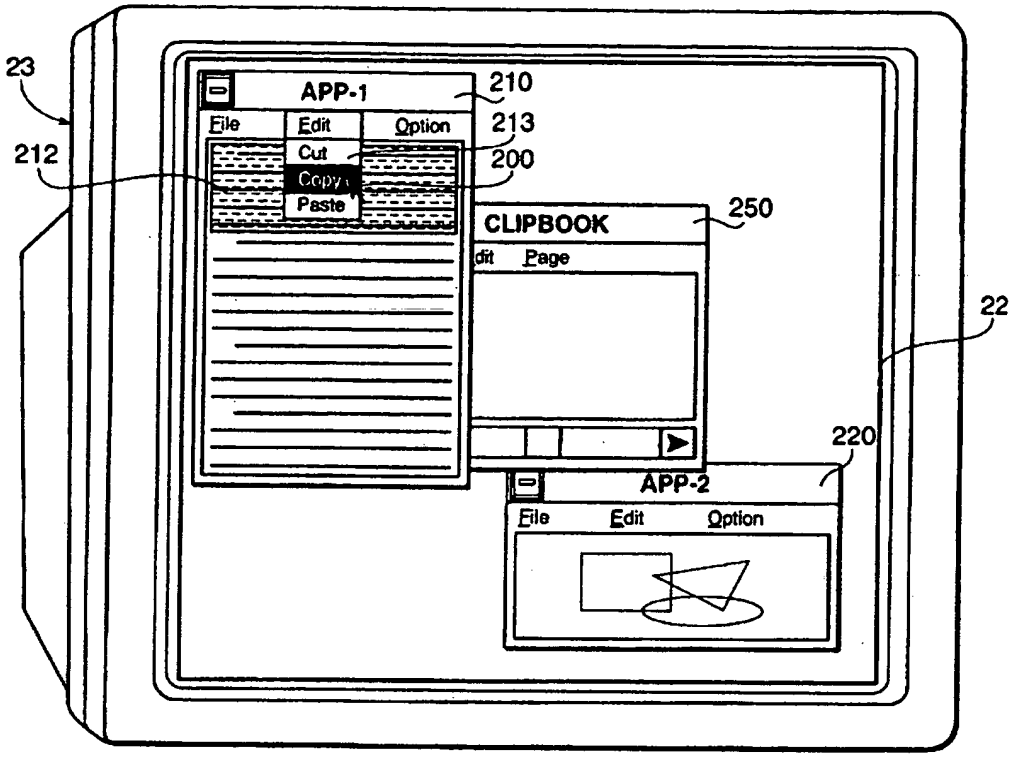


FIG. 14(c) PRIOR ART

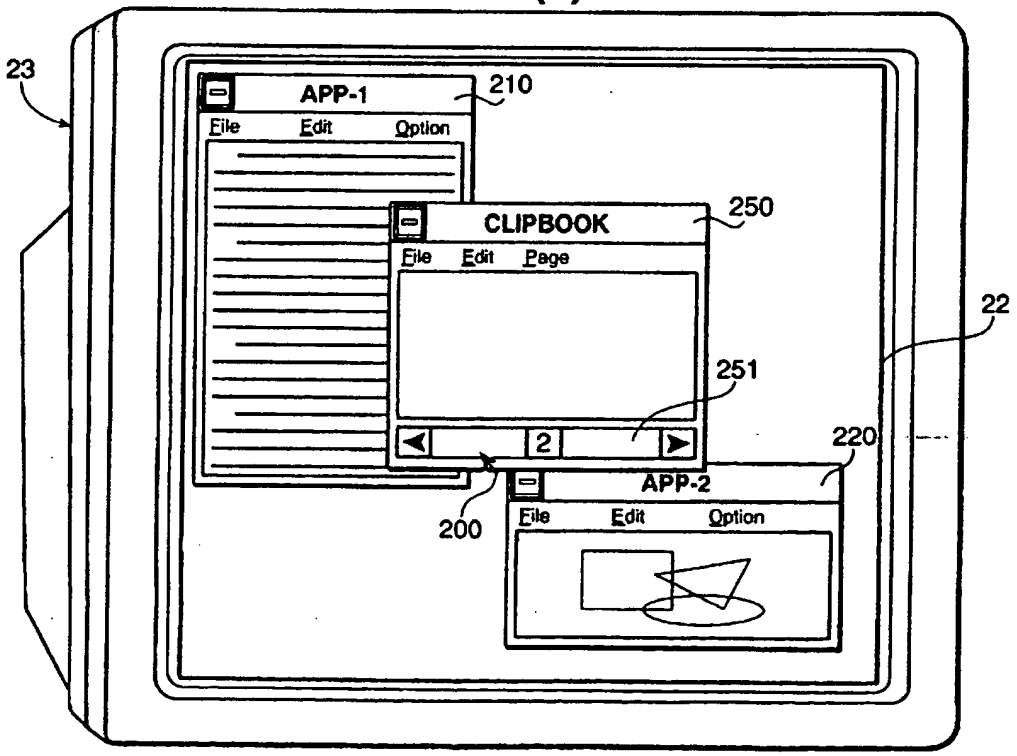


FIG. 14(d) PRIOR ART

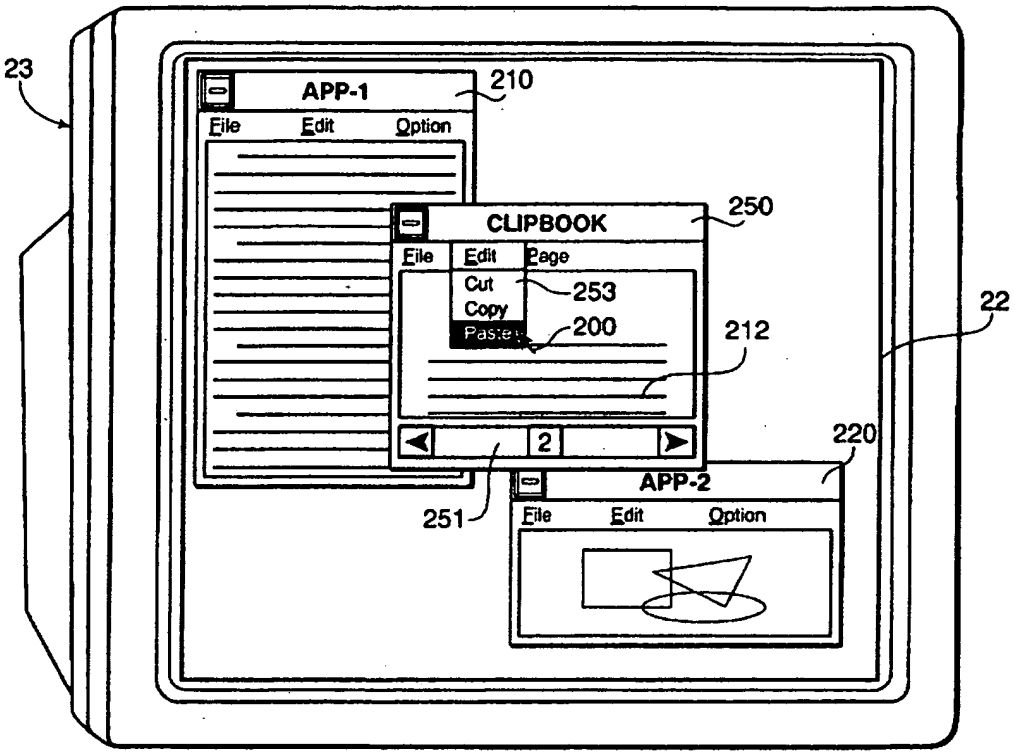


FIG. 14(e) PRIOR ART

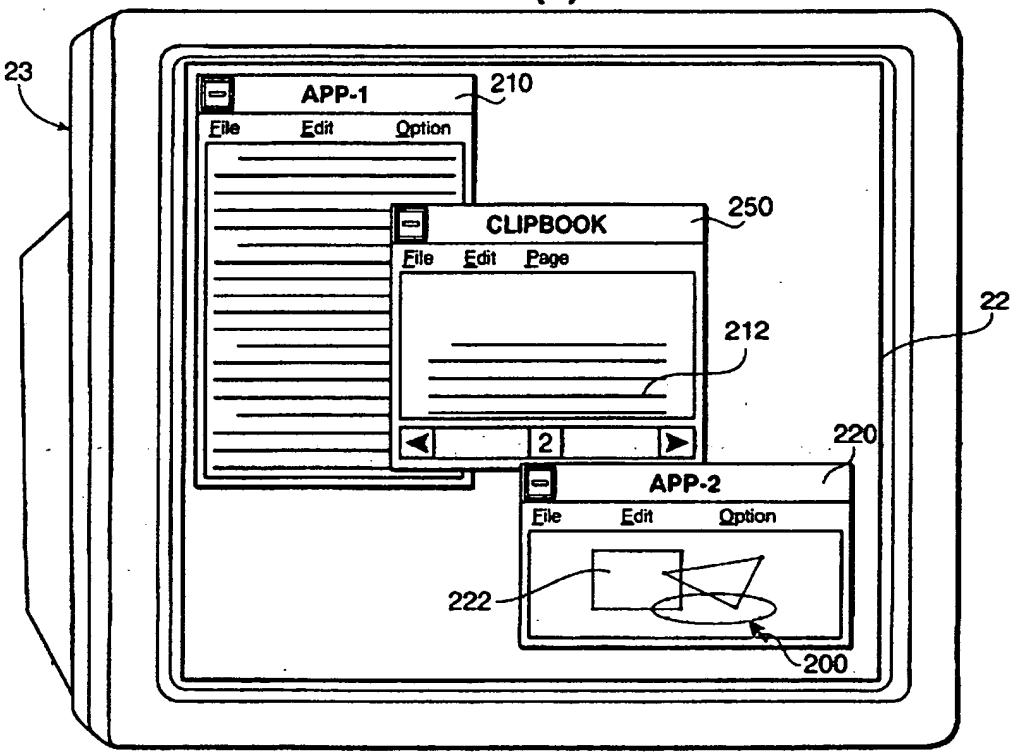


FIG. 14(f) PRIOR ART

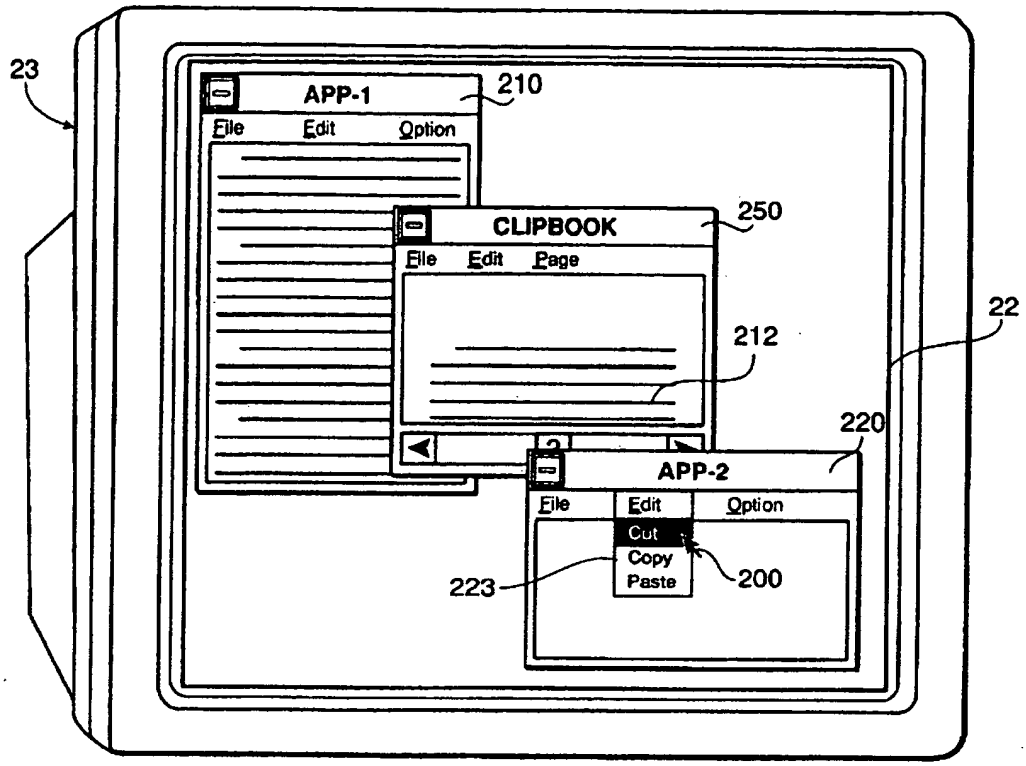


FIG. 14(g) PRIOR ART

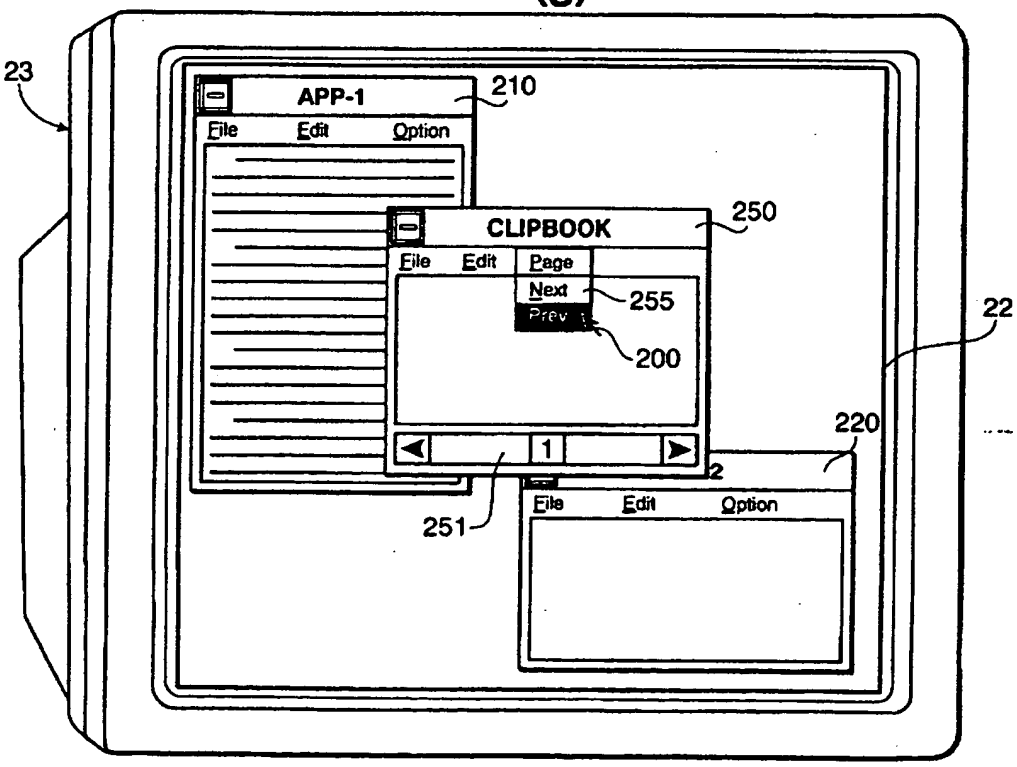


FIG. 14(h) PRIOR ART

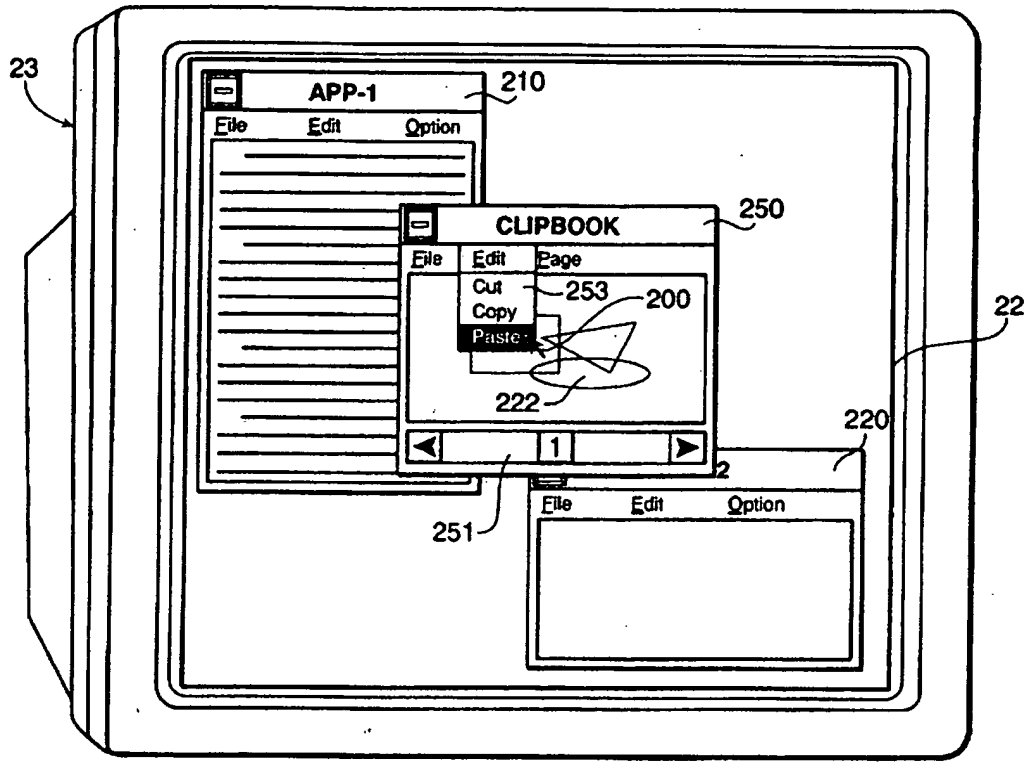


FIG. 14(i) PRIOR ART

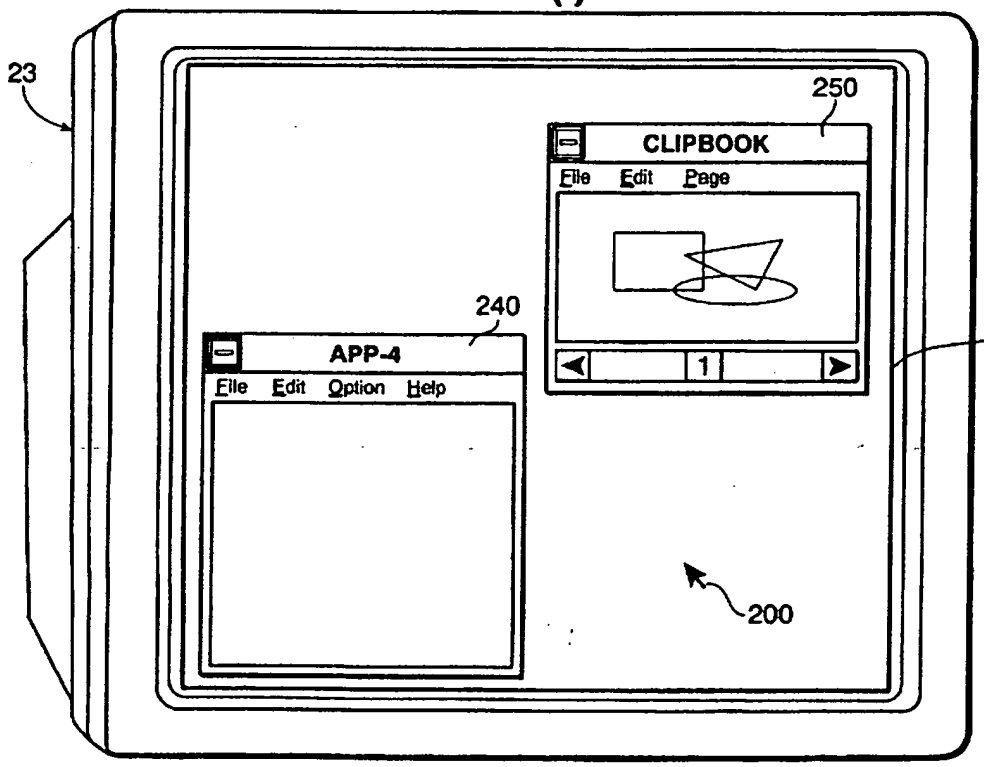


FIG. 14(j) PRIOR ART

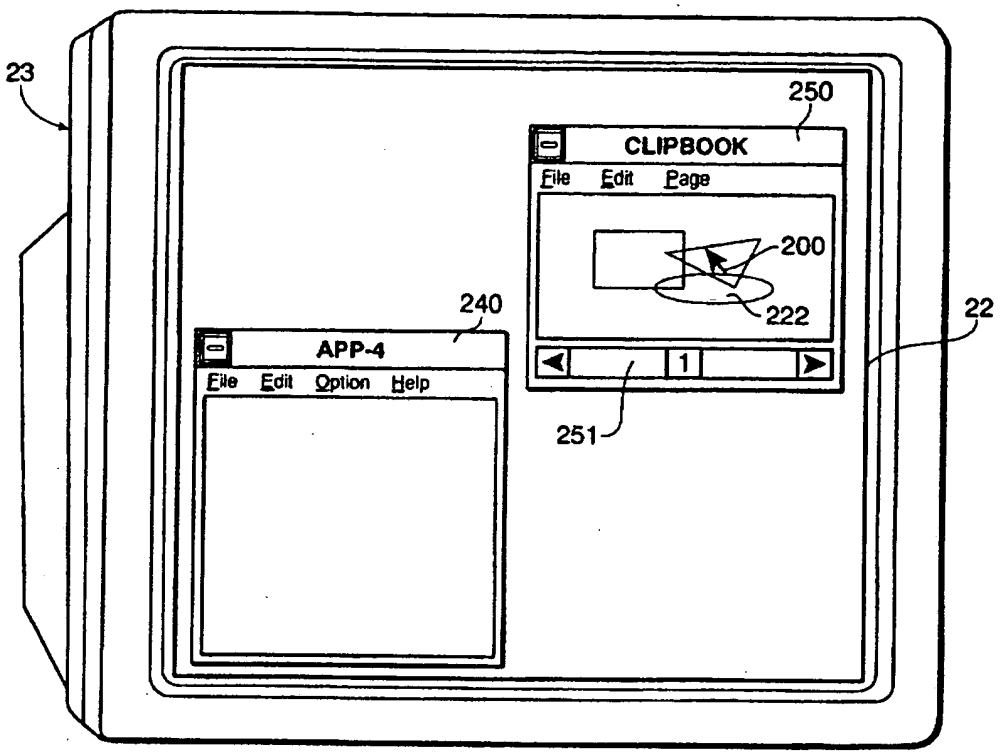


FIG. 14(k) PRIOR ART

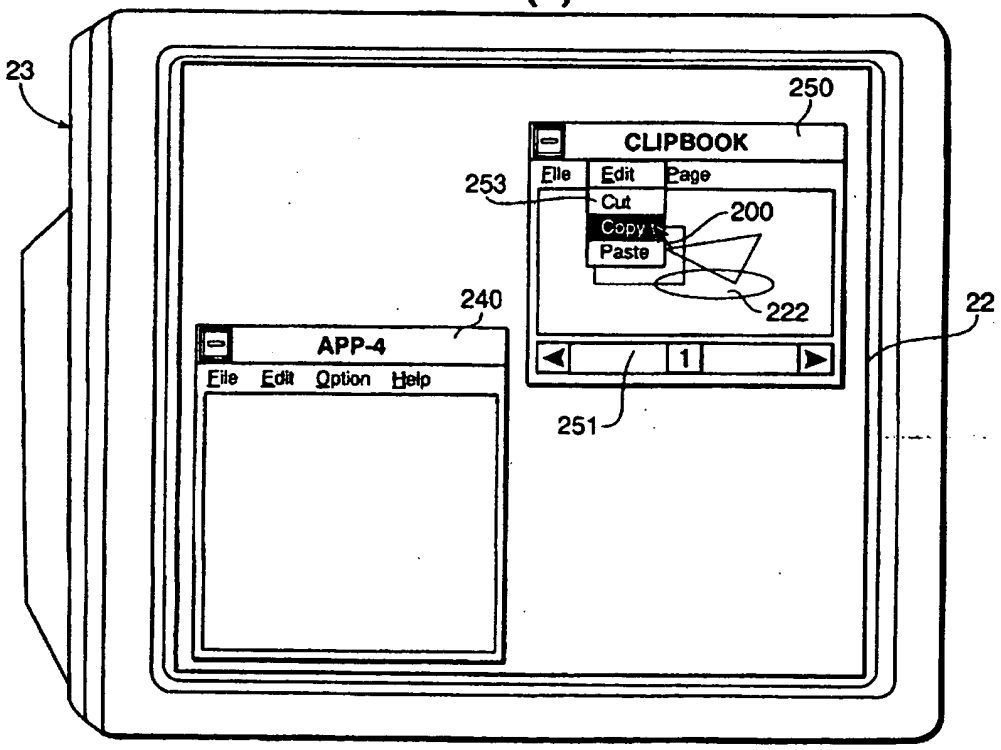


FIG. 14(l) PRIOR ART

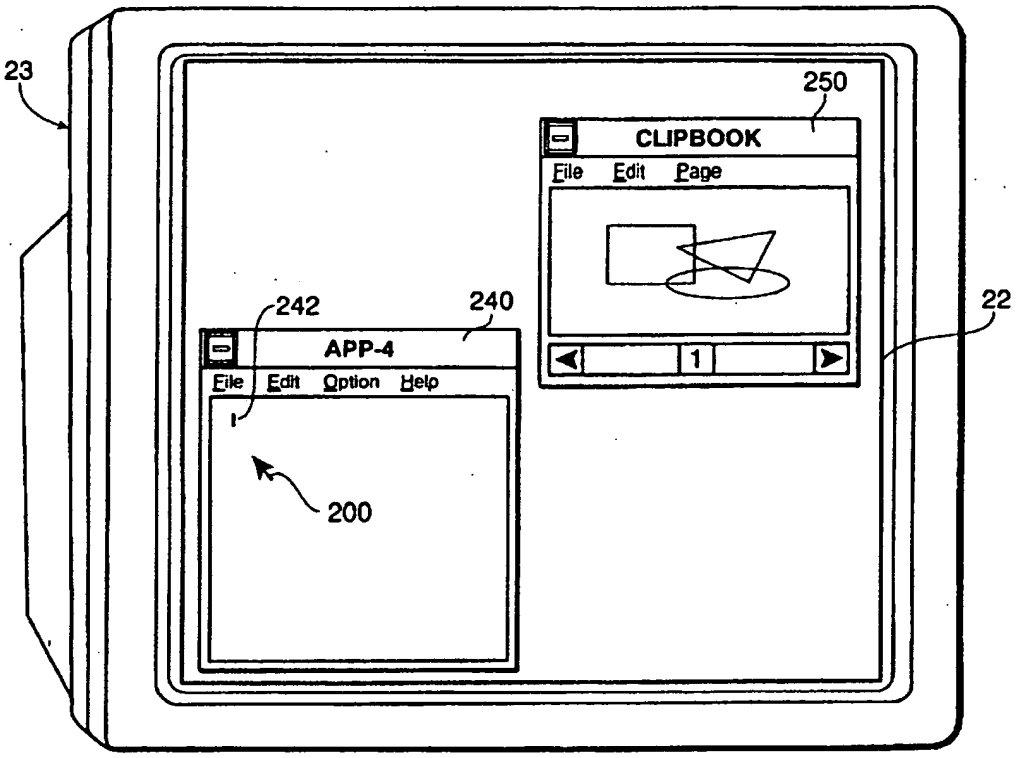


FIG. 14(m) PRIOR ART

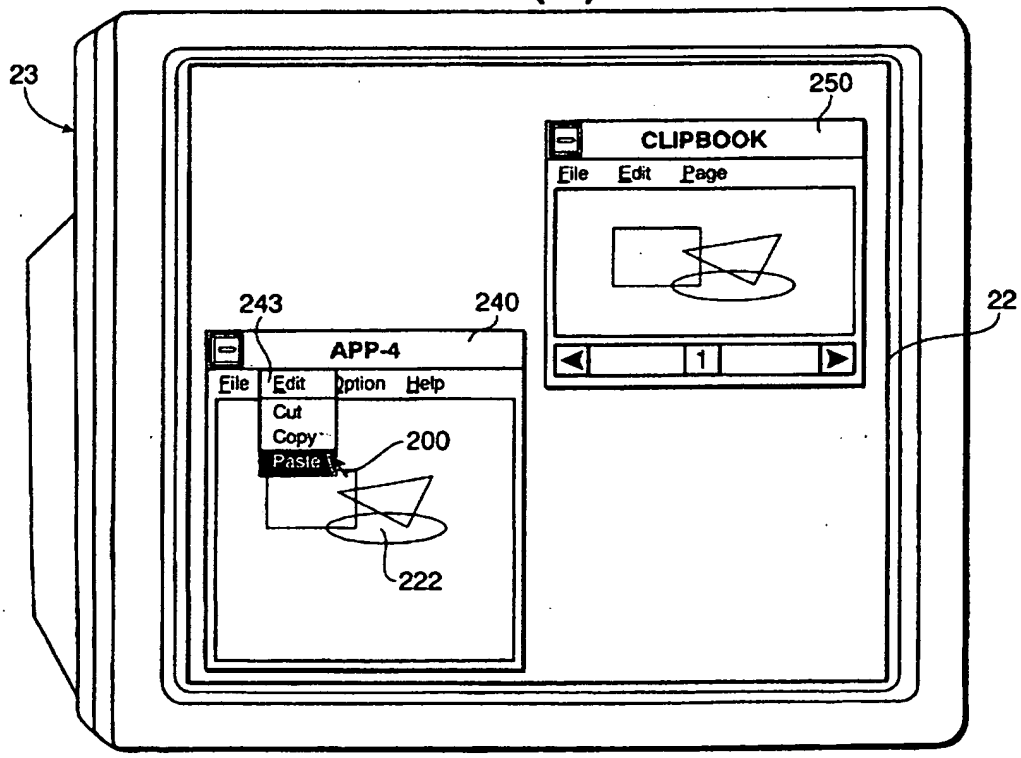


FIG. 14(n) PRIOR ART

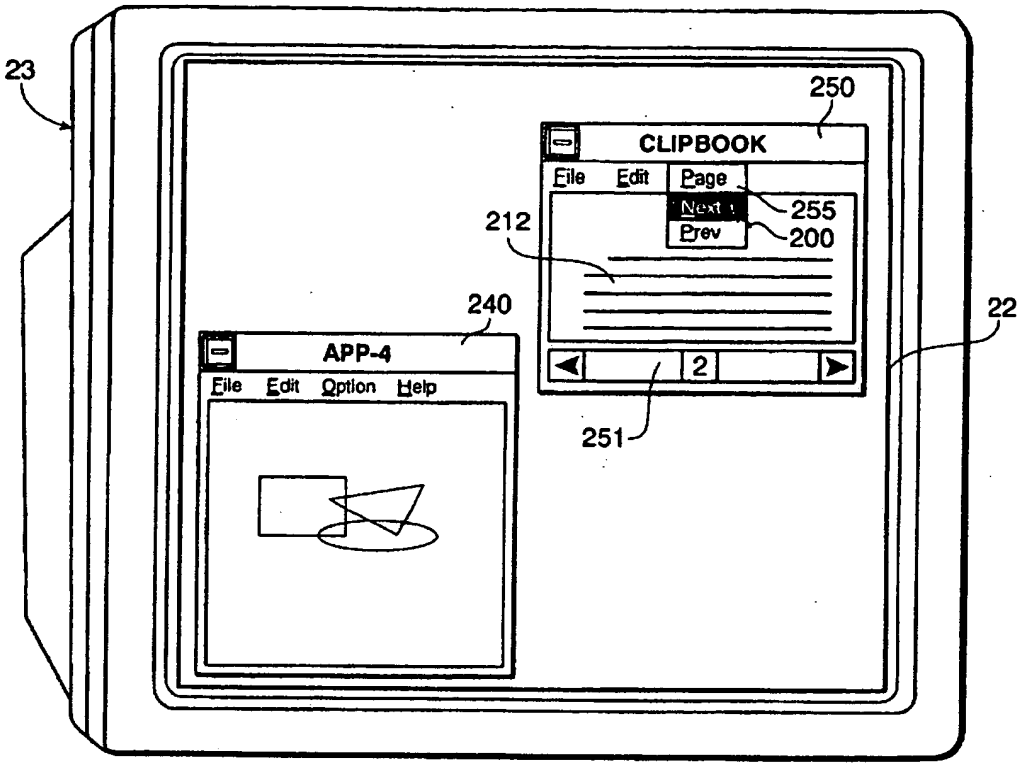


FIG. 14(o) PRIOR ART

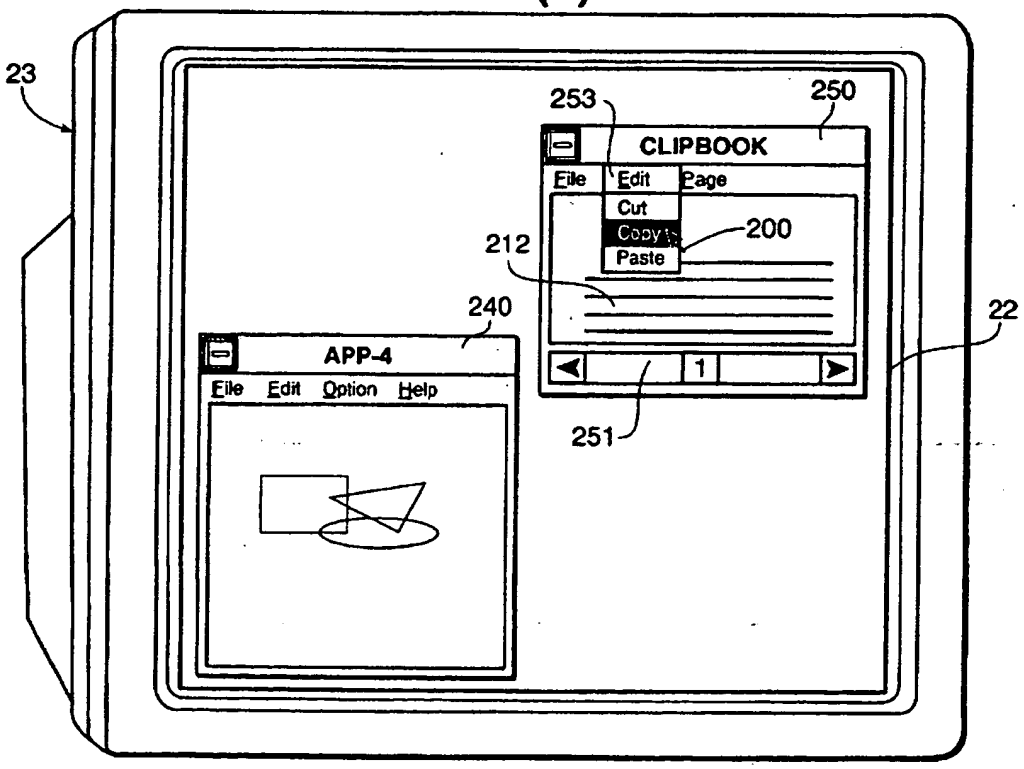


FIG. 14(p) PRIOR ART

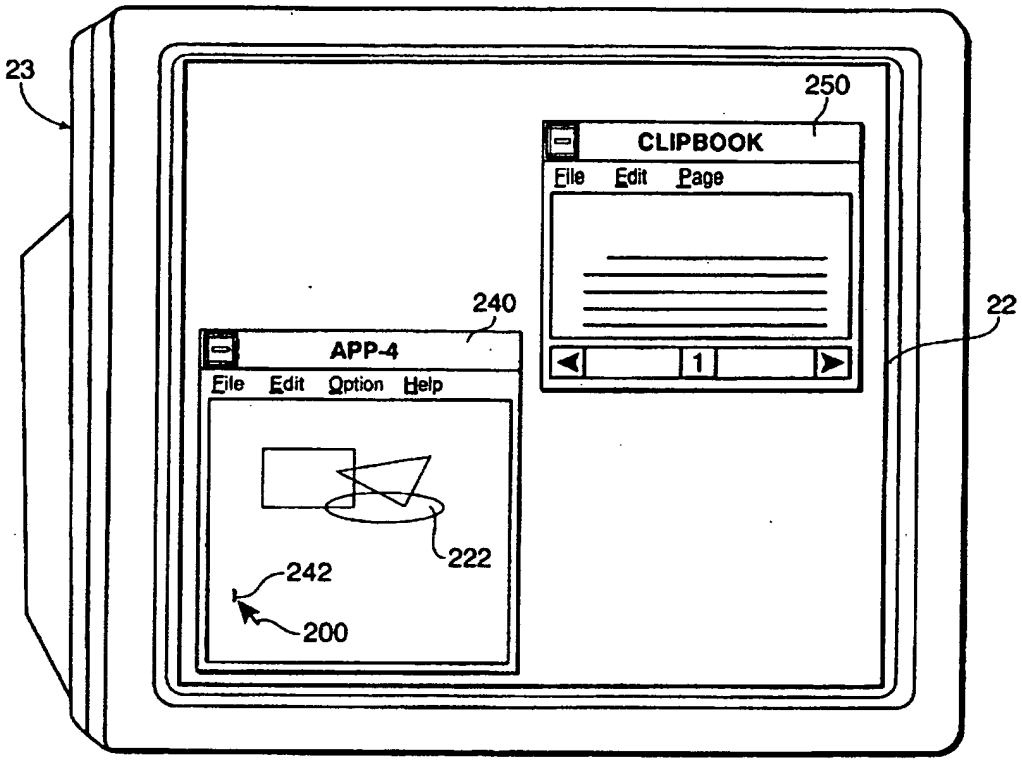


FIG. 14(q) PRIOR ART

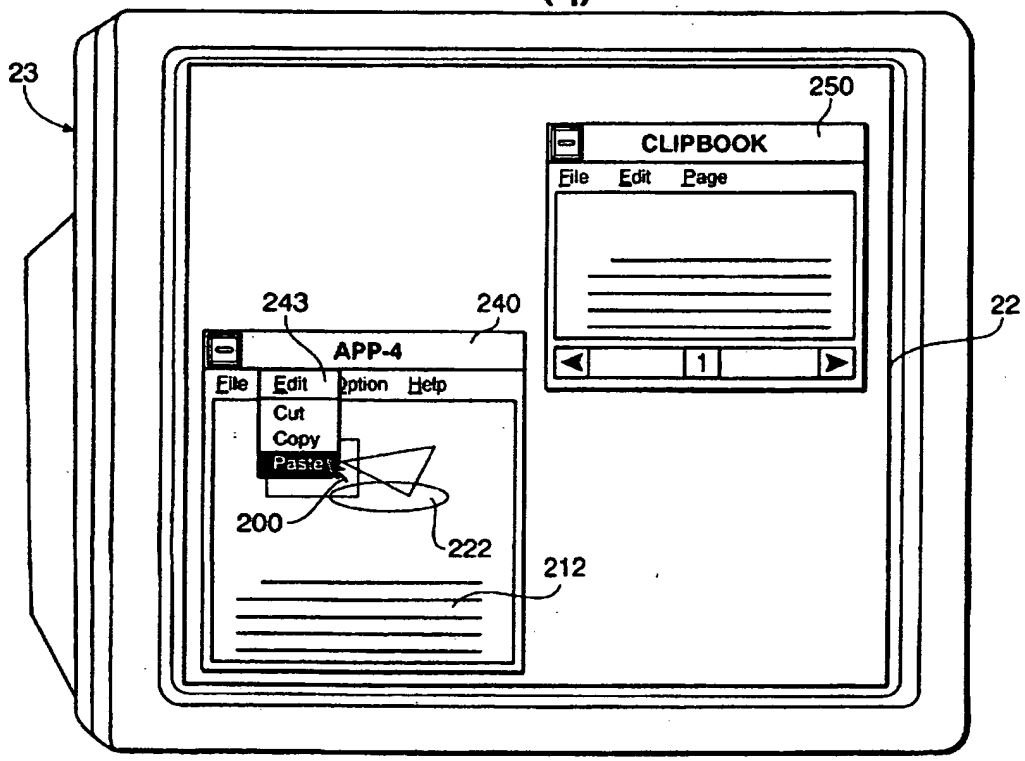


FIG. 14(r) PRIOR ART

DATA TRANSFER SYSTEM

BACKGROUND OF THE INVENTION

1. Field Of The Invention

The present invention concerns a computer-implemented data transfer system which, in a windowing environment, manages multiple memory buffers so as to permit data transfer within or between different application programs. More particularly, the invention allows a user to store multiple data selections into different ones of the multiple memory buffers without leaving focus from an instance of an application program.

2. Description of the Related Art

One advantage of windowing environments is the ability to transfer data easily from one application program to another. A user simply launches both applications (multitasking being another advantage of windowing environments), sets focus on the first application and selects data in that application, stores the selected data to a memory buffer, switches focus to the second application, and recalls the data from the memory buffer to the second application.

Conventional windowing environments, such as that provided by Microsoft Window's version 3.1 ("Windows" hereinafter), have only a single memory buffer for transferring data within and between different application programs. In Windows, this buffer is called the clipboard. Because there is only a single memory buffer, whenever data is transferred into the buffer, any old data stored there is lost. This arrangement makes it difficult for a user to transfer multiple data selections, since to avoid data loss a transfer operation for each data selection must be completed before each subsequent data selection can be made.

For example, in order to transfer two data selections from a first application program to a second application program, the following steps are necessary: (1) the first application program must be brought into focus; (2) the first data selection in the first application program must be selected; (3) the data selection must be stored into the clipboard; (4) the second application program must be brought into focus; (5) the position where the data is to be transferred to must be selected; (6) the data must be recalled from the clipboard to the selected position in the second application program; (7) the first application program must be brought back into focus; (8) the second data selection must be selected in the first application program; (9) the second data selection must be stored to the clipboard; (10) the second application program must be brought back into focus; (11) the position where the second data selection is to be transferred to must be selected; and (12) the second data selection must be recalled from the clipboard to the selected position in the second application program.

This cumbersome process is a needless waste of the user's time and efforts, both in terms of the number of steps involved and in terms of the number of times that the user must shuffle between various windows. In addition, the user is restricted to dealing with a single data selection at a time. Since a user can not deal with multiple data selections at the same time, the user can easily forget which one of the multiple data transfers he is currently trying to make, as well as forget how far in a current data transfer he has already progressed.

"Clipbooks" or "scrapbooks" are known attempts to address the difficulties encountered in copying multiple data selections. Clipbooks or scrapbooks are application programs that contain multiple pages, each of which can store one data selection, so as to allow a user to make multiple data selections without losing prior data selections. However, after each data selection, it is necessary to shift focus away from the application from which data is copied, so as to permit designation of one of the multiple pages onto which the data is stored. This is illustrated in the following example with reference to FIGS. **14**(*a*) to **14**(*r*). In connection with these figures, and throughout the remainder of this specification, the terms "copy" and "paste", as well as "cut" and "paste", are given their conventionally accepted meanings, as defined, for example, in F. Davis, "The Windows 3.1 Bible", Peachpit Press, 1993.

FIG. **14**(*a*) shows a computer display with two application programs, APP-1 **210** and APP-2 **220**, clipbook **250**, and cursor **200**. In FIG. **14**(*b*), APP-1 **210** is brought into focus, as shown by the double border around its window, and text data selection **212** is selected, as depicted by the thick border around the five lines at the top of APP-1 **210**. In FIG. **14**(*c*), text data **212** is copied into the clipboard (not shown) by use of edit pull-down menu **213**. In FIG. **14**(*d*), clipbook **250** is brought into focus, as indicated by the double border surrounding its window. In addition, scroll bar **251** is used to select page 2 in clipbook **250**. In FIG. **14**(*e*), text data selection **212** is pasted into page 2 of clipbook **250** by use of edit pull-down menu **253**. In FIG. **14**(*f*), APP-2 **220** is brought into focus and graphic data selection **222** is selected. In FIG. **14**(*g*), graphic data selection **222** is cut into the clipboard by use of edit pull-down menu **223**. (As is known, because the data is "cut" rather than "copied" into the clipboard, graphic data selection **222** is deleted from APP-2 **220**.) In FIG. **14**(*h*), clipbook **250** is brought into focus and page 1 of clipbook **250** is selected by page pull-down menu **255**. In FIG. **14**(*i*), graphic data selection **222** is pasted into page 1 of clipbook **250** by use of edit pull-down menu **253**.

At this point, two data selections, text data **212** and graphic data **222**, are copied into two different pages of clipbook **250**. Now, the user can close APP-1 **210** and APP-2 **220** and open another application into which data selections can be transferred, as described below in connection with FIGS. **14**(*j*) through **14**(*r*).

FIG. **14**(*j*) shows newly opened APP-4 **240** and clipbook **250**. In FIG. **14**(*k*), clipbook **250** is brought into focus and graphic data selection **222** is selected. In FIG. **14**(*l*), graphic data selection **222** is copied from page 1 of the clipbook into the clipboard (not shown) by use of edit pull-down menu **253**. In FIG. **14**(*m*), APP-4 **240** is brought into focus and position **242** in APP-4 **240** is selected with cursor **200**. In FIG. **14**(*n*), graphic data selection **222** is pasted into APP-4 **240** by use of edit pulldown menu **243**. In FIG. **14**(*o*), clipbook **250** is brought back into focus and page 2 of clipbook **250** is selected with page pull-down menu **255**. In FIG. **14**(*p*), text data selection **212** is copied into the clipboard (not shown) by use of edit pull-down menu **253**. In FIG. **14**(*q*), APP-4 **240** is brought back into focus and position **242** is selected. Finally, in FIG. **14**(*r*), text data selection **212** is pasted into APP-4 **240** by use of edit pull-down menu **243**.

Thus, even though clipbooks and the like allow a user to make multiple data selections without losing data from a prior selection, because a user must shift focus away from the application after each data selection, use of such clipbooks is a cumbersome and unnecessarily lengthy procedure.

Accordingly, there exists a need to allow a user to make multiple data selections without leaving focus from a single application, and to allow a user to transfer multiple data selections within and between applications.

SUMMARY OF THE INVENTION

These needs are addressed by the present invention by providing a data transfer system with multiple memory buffers, each of which can be selected without leaving focus from a currently-active application. By virtue of this operation, the user can move multiple data selections within and between application programs without needless shuffling between the application programs and their respective windows, saving the user considerable time and effort.

According to one aspect of the present invention, a computer implemented method for transferring data from a first application program into at least one of a plurality of memory buffers comprises the steps of bring the application program into focus, selecting data in the application program, selecting a "copy" or a "cut" operation while the application program is still in focus, selecting one of the memory buffers while the application program is still in focus, and automatically storing the selected data into the selected one of the memory buffers. Multiple "copy" or "cut" operations can be made without ever leaving focus of the application program.

Thereafter, data can be recalled from any one of the plurality of memory buffers into a second application program by bringing the second application program into focus, selecting a position in the second application program (if necessary), selecting a "paste" operation while the second application program is still in focus, selecting one of the memory buffers while the second application program is still in focus, and automatically recalling data from the selected one of the memory buffers to the selected position in the second application program. Multiple "paste" operations can be made without ever leaving focus of the second application program.

In another aspect of the present invention, a method for transferring data from a first application to a second application using one of a plurality of memory buffers comprises the steps of bringing the first application into focus, selecting data from the first application, selecting one of the plurality of memory buffers while the first application remains in focus, storing the selected data to the selected one of the plurality of editing buffers, bringing the second application into focus, selecting a position for transferring data into the second application (if necessary), re-selecting the selected one of the plurality of memory buffers while the second application is in focus, and recalling data from the re-selected one of the plurality of memory buffers to the selected position in the second application.

This brief summary has been provided so that the nature of the invention may be understood quickly. A more complete understanding of the invention can be obtained by reference to the following detailed description of the preferred embodiment thereof in connection with the attached drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** is a view showing the outward appearance of representative computing equipment.

FIG. **2** is a detailed block diagram showing the internal construction of computing equipment as shown in FIG. **1**.

FIG. **3** is a functional block diagram used to explain messaging between a windowing operating system and multiple applications and multiple memory buffers in a data transfer system using multiple memory buffers.

FIG. **4**(*a*) is a flow chart showing operation of the data transfer system during a "cut" or "copy" operation when storing data from an application program to one of multiple memory buffers.

FIG. **4**(*b*) is a flow chart showing the operation of the data transfer system during a "paste" operation when recalling data from one of multiple memory buffers into an application program.

FIGS. **5** to **11**(*b*) show one representative operation of the data transfer system using multiple memory buffers in transferring multiple data selections from two application programs to two other application programs from a user's point of view.

FIGS. **12**(*a*) and **12**(*b*) are flow charts showing, from a user's point of view, the operation of the data transfer system using multiple memory buffers to copy data from an application program to one of the memory buffers.

FIGS. **13**(*a*) and **13**(*b*) are flow charts showing the operation of the data transfer system using multiple memory buffers to copy data from one of the memory buffers to an application program, again from a user's point of view.

FIGS. **14**(*a*) to **14**(*r*) show prior art processes for transferring multiple data selections between application programs using a clipbook as a temporary storage buffer for the multiple data selections.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

FIG. **1** is a view showing the outward appearance of representative computing equipment incorporating the invention and which can be used for many purposes, possibly including word processing, graphics design, spreadsheets, etc.

Shown in FIG. **1** is computing equipment **20** such as an IBM-PC or PC compatible computer having a windowing operating system such as Windows. Computing equipment **20** is provided with monochromatic or color display monitor **23** having display screen **22**, on which computing equipment **20** displays images to the user. Computing equipment **20** is further provided with floppy disk drive **24** with which removable disk media may be read or written. A CD-ROM driver can be used in place of or in addition to floppy disk drive **24**.

Computing equipment **20** is also equipped with fixed disk drive **25** as one possible medium for storing the windowing operating system as well as data files and application programs, keyboard **26** for permitting input of text data and manipulation of objects displayed on display screen **22**, and pointing device **27**, such as a mouse, which is provided for pointing and for manipulating objects displayed on display screen **22**. If desired, printer **30** may be provided for outputting documents processed by computing equipment **20**.

Also provided are conventional connections to other data and application storage media, such as connection **31** to a network.

In accordance with user instructions, and under control of the windowing operating system, stored application programs such as desktop publishing application programs, drawing application programs, word processing application programs, and the like, are selectively activated to process and to manipulate data.

FIG. **2** is a detailed block diagram showing one possible internal construction of computing equipment **20**. As shown in FIG. **2**, computing equipment **20** includes a central processing unit (CPU) **40** such as a programmable microprocessor interfaced to computer bus **41**. Also interfaced to

computer bus **41** are display interface **44**, network interface **45**, printer interface **47**, floppy disk drive **24**, keyboard **26**, pointing device **27**, main memory **51**, read only memory (ROM) **52**, and fixed disk **25**. Other interfaces may be provided, as appropriate to the overall configuration of computing equipment **20**.

Main memory **51**, such as random access memory (RAM), interfaces to computer bus **41** so as to provide CPU **40** with access to memory storage. In particular, when executing stored application program instruction sequences such as those associated with application programs stored on fixed disk **25**, CPU **40** loads those instruction sequences from fixed disk **25** (or other storage media such as media accessed via network interface **45** or floppy disk drive **24**) into main memory **51** and executes those stored program instruction sequences out of main memory **51**.

ROM **52** is provided for storing invariant instruction sequences such as start-up instruction sequences of basic input/output operating system (BIOS) sequences for operation of keyboard **26**.

As shown in FIG. **2**, fixed disk **25** stores program instruction sequences for windowing operating systems and various application programs, document files, and other data files. Fixed disk **25** can also store any number of other types of files.

[Windowing Operating System]

A user interacts with application programs by means of the windowing operating system. This operating system is used to start instances of application programs, to interact with those application programs, and to close instances of application programs.

An instance of an application program is simply one copy of an application program that is running at one time. Multiple instances of a single application program can be run at once. Each instance is typically represented by a window or windows. Each instance of an application program has its own data, and is a separate entity from the perspective of the user.

[Data Transfer System]

FIG. **3** is a functional block diagram showing the interactions between user inputs, window operating system **100**, applications **104**, and a "copystack" memory **106** for a data transfer system with multiple memory buffers for storing data selected from an application to a selected one of the multiple memory buffers. Preferably, the number of memory buffers in copystack **106** is automatically increased, up to some predetermined number such as **20**, by the windowing operating system **100** so that there is at least one blank memory buffer into which new data can be stored. The memory buffers are system-wide buffers in that they can be accessed by any application program running under windowing operating system **100**.

In FIG. **3**, the copystack and data transfer system are treated as part of the windowing operating system. It should be understood, however, that other arrangements are possible. For example, one other possible implementation is to provide multiple system wide memory buffers separate from the windowing operating system, such as in an add-on utility or through use of OLE ("object linking and embedding") 2.0 protocol. As another possible implementation, the multiple memory buffers might be included by pre-arrangement in a suite of software applications.

In FIG. **3**, the data transfer system involves interaction among windowing operating system **100**, applications **104**, and copystack **106** for multiple memory buffers. The user input **102** is via a graphical user interface (GUI).

FIG. **4**(*a*) is a flow diagram illustrating execution of stored program instruction sequences by which the data transfer system operates during a "cut" or "copy" operation so as to store data from an application program to a user-selected one of multiple memory buffers. Thus, in response to user designation of a "cut" or "copy" operation in step **S401**, flow advances to step **S402** in which the active application sends a message (depicted at **402** in FIG. **3**) to operating system **100** advising the operating system that the user has selected a "cut" or "copy" operation, and requesting operating system **100** to provide data needed by the application so as to effectuate the "cut" or "copy" operation. Such data might include, for example, the location of a memory buffer into which data from the application should be stored, as well as data needed by the application so as to draw an appropriate menu for the user to select desired parameters (such as total number of memory buffers in the copystack, type of data stored in each memory buffer, and the like) for the "cut" or "copy" operation. Operating system **100** responds in step **S403** by sending a message to copystack **106** (depicted at **403** in FIG. **3**) advising copystack **106** that a memory buffer is needed for a "cut" or "copy" operation. Copystack **106** responds by reserving memory for a new memory buffer and, in step **S404**, sending a message (depicted at **404** in FIG. **3**) to operating system **100** relaying copystack information such as the location of the new memory buffer, the total number of memory buffers now available for a "copy" or "cut" operation, and the kind of data (such as text, graphic, spreadsheet, etc) in each of the different memory buffers. In step **S405**, the operating system sends a message (depicted at **405** in FIG. **3**) to the application so as to relay the copystack information. At step **S406**, the application stores data selected by the user to a user-specified copystack memory buffer (or, in the absence of user specification, to the default copystack memory buffer), whereafter data is optionally deleted at step **S408** in dependence on whether a user has specified a "cut" operation or a "copy" operation (step **S407**).

FIG. **4**(*b*) is a flow diagram illustrating the stored program instruction steps executed by the data transfer system during a "paste" operation so as to recall data from one of the multiple memory buffers in copystack **106** to an active application program.

Thus, in response to a user selection of a "paste" operation (step **S410**), flow advances to step **S411** in which the active application sends a message (depicted at **411** in FIG. **3**) to operating system **100** advising operating system **100** of the "paste" operation. In step **S412**, operating system **100** sends a message (depicted at **412** in FIG. **3**) to copystack **106** requesting for copystack **106** to provide copystack information. Copystack **106** responds in step **S413** by sending a message (depicted at **413** in FIG. **3**) to operating system **100** with the needed copystack information, namely the number and locations of memory buffers currently being managed by copystack **106**, as well as the type of data stored in each memory buffer, the identity of the last-accessed memory buffer and the like. Flow advances to step **S414** in which operating system **100** sends a message (depicted at **414** in FIG. **3**) to the active application relaying copystack information. Based on the copystack information so provided, the active application draws an appropriate menu for the user to designate parameters for the "paste" operation, such as which of the multiple memory buffers to recall data from. In step **S416**, after selection of one of the multiple memory buffers (which may be by user-selection or by default selection such as the last-accessed memory buffer), data is recalled from the selected memory buffer to memory which is accessible by the active application. Thereafter, in step **S417**, the selected memory buffer is closed.